THE UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR E.J.BAETS

TITLE OF THESIS MULTIVARIABLE SELF-TUNING REGULATORS

DEGREE FOR WHICH THESIS WAS PRESENTED MASTER OF SCIENCE

YEAR THIS DEGREE GRANTED Fall, 1980

THE UNIVERSITY OF ALBERTA

MULTIVARIABLE SELF-TUNING REGULATORS

by

© E.J.BAETS

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE

OF MASTER OF SCIENCE

IN

PROCESS CONTROL (CHEMICAL ENGINEERING)

DEPARTMENT of CHEMICAL ENGINEERING

EDMONTON, ALBERTA

Fall, 1980

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research, for acceptance, a thesis entitled MULTIVARIABLE SELF-TUNING REGULATORS submitted by E.J.BAETS in partial fulfilment of the requirements for the degree of MASTER OF SCIENCE in PROCESS CONTROL (CHEMICAL ENGINEERING).

ABSTRACT

A multivariable self-tuning controller based on least squares estimation and minimum variance control is described. As for single input - single output systems some modifications can be made in order to improve the algorithm's performance for certain types of process system dynamic behaviour. Exponential forgetting factors can increase the rate of convergence of the parameters. Implementation of integral control action prevents offset and feedforward compensation can improve the output behaviour after disturbances if the relation between output and disturbance is linear. Using the average output as an input for the controller instead of the instantaneous output ensures a smaller average error if the process is disturbed or if the signal-to-noise ratio is small.

The self-tuning controller and the modifications described above are tested out by a series of simulations of linear multivariable systems and simulation of the control behaviour of a distillation column characterized by a transfer function model. Simulation of the control behaviour of a binary distillation column with its dynamic behaviour described by a nonlinear differential equation model showed that the multivariable self-tuning controller is well suited to handle nonlinearities and interaction in a process. However, the applicability of the controller is limited by its inability to deal with large time delays in one loop of a multivariable process.

ACKNOWLEDGEMENTS

The author wishes to thank her thesis supervisors Dr. R.K. Wood and Dr. S.L. Shah for their guidance and assistance in this project. Thanks also go to the staff of the DACS centre in the Department of Chemical Engineering who aided in the use of the computing facilities, and to the graduate students for their advice and help.

The author also acknowledges Rotary International, whose financial assistance made possible this project.

Finally, the author wishes to thank her parents for their encouragement and support.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

FIGURE PAGE

NOMENCLATURE

A_i	parameter associated with the output in a single input - single output process model
$\underline{\underline{A}}_i$	parameter matrix associated with the output in a multivariable process model
$\underline{\underline{A}}i$	parameter matrix associated with the output in a multivariable prediction model
AIE	accumulated variance
B_i	parameter associated with the input in a single input - single output process model
$\underline{\underline{B}}_i$	parameter matrix associated with the input in a multivariable process model
$\underline{\underline{B}}i$	parameter matrix associated with the input in a multivariable prediction model
$\underline{\underline{C}}_i$	parameter matrix in a process noise model
$\underline{e}$	vector of white noise elements
$\underline{\underline{F}}$, $\tilde{\underline{\underline{F}}}$	matrices in minimum variance controller
$\underline{\underline{G}}$, $\tilde{\underline{\underline{G}}}$	matrices in minimum variance controller
$\underline{\underline{G}}$	parameter matrix associated with measurable disturbance
G_{ij}	transfer function between output i and input j
$\underline{\underline{I}}$	unity matrix
k	time delay in discrete process model
$\underline{\underline{K}}$	gain matrix in self-tuning regulator algorithm
L_i	disturbance transfer function
n	number of matrices associated with output in discrete process model
N	number of estimated A matrices
m	number of matrices associated with input in discrete process model

M	number of estimated B matrices
p	dimension of a process
$\underline{\underline{P}}$	covariance matrix of estimated parameters and matrix in self-tuning regulator algorithm
q	forward shift operator
$\underline{\underline{Q}}$	positive semidefinite matrix
RAFT	required average over finite time
STR	self-tuning regulator
t	time as a function of the sampling interval
T	number of sampling intervals
$\underline{u}$	vector of process input signals
v	average output error
$\underline{\underline{V}}$	matrix of errors
$\underline{y}$	vector of output signals
y_a	average output
y_o	optimal output value
y_r	reference value for the output
y_s	set point for the output
$\underline{z}$	vector of process disturbance signals
α_i	parameter associated with output signal in estimation equation of single input - single output STR
$\underline{\underline{\alpha}}i$	parameter matrix associated with output signals in estimation equation of multivariable STR
β_i	parameter associated with input signal in estimation equation of single input - single output STR
$\underline{\underline{\beta}}i$	parameter matrix associated with input signals in estimation equation of multivariable STR
$\underline{\underline{\gamma}}^i$	parameter matrix in estimation equation

	associated with disturbance model
Δ	difference
$\underline{\varepsilon}$	white noise
$\underline{\underline{\theta}}$	matrix of parameter estimates
μ	eigenvalue
$\underline{\psi}$	vector of output and input measurements
$\underline{\underline{\Psi}}$	matrix of output and input measurements
σ	noise covariance factor
ω	exponential forgetting factor
$\hat{}$	estimated value

1. INTRODUCTION

The design of a control scheme for a process is usually based on a mathematical model of the system. This model can be obtained by analyzing the process and setting up a number of equations, or by an identification method. The second method is only possible if the process is in an operating state and if enough information can be obtained from measurements of the inputs and outputs. Because most systems are nonlinear, a linear mathematical model only describes the process for one set of operating conditions. Therefore, it is necessary in most cases to update the model from time to time and to tune the control parameters.

Aström and his co-workers [4] developed a technique which combines identification and control and which has been designated as self-tuning regulators. The recursive algorithm which estimates the parameters of a process model provides the control algorithm with the information for tuning the control parameters.

One of the obvious advantages is that even if the process parameters change during operation, the controller will automatically adapt. However, there are also a few disadvantages. Self-tuning regulators of the type described here do not work for nonminimum phase systems. Also, the algorithm is much larger than for basic controllers such as PID, and therefore a larger computer installation is required. Although the algorithm estimates the parameters of

a process model, certain parameters must be initially provided which must be related to the process. Thus, a good knowledge of the system is still necessary.

There are many different methods of identification and control. Aström based his self-tuning regulator on the techniques of least squares estimation and minimum variance control [1]. This provides good results for a certain class of systems, but there are two main disadvantages. First, the control technique does not provide good closed loop stability for nonminimum phase systems. Although a suboptimal technique can be used in such a case, this assumes that one knows that the process is nonminimum phase. Second, the input variance can become fairly large, which is undesirable for practical reasons.

Different schemes were proposed in order to improve the technique. Clarke and Gawthrop [8] developed a method which incorporates the system input and the set point, as well as the system output, in the cost function. This will reduce the input variance. Wellstead, Prager and Zanker [22] proposed a pole assignment self-tuning regulator. Such a regulator can effectively deal with nonminimum phase systems. A few successful applications of self-tuning regulators have been reported.

Previously only a few papers dealt with multivariable self-tuning controllers; for example, Borisson [7] recently extended Aströms basic algorithm to the multivariable case. In this work the multivariable self-tuning controller is

investigated. Some modifications are made to improve the closed loop performance under certain conditions. Finally, it will be applied to a nonlinear simulation model of a binary distillation column.

2. MULTIVARIABLE SELF-TUNING REGULATORS

2.1 Introduction to multivariable self-tuning regulators

A process is called multivariable if there is more than one measurable output and if an input influences more than one output, i.e. if there is interaction.

In this study only multivariable systems with an equal number of inputs and outputs are considered. For a linear system, each output variable can be associated with each input variable, and thus a 2 x 2 system can be represented by a transfer function model as shown in Figure 1. The most suitable mathematical representation is in matrix form.

Linear quadratic control theory gives a general method for the design of regulators for multivariable systems if a mathematical model of the system is available. The control strategy is obtained by solving a Riccati equation. If the control strategy has to be computed at every sampling instant, as is the case in self-tuning regulators, it is desirable to have an algorithm that is easier to solve than a Riccati equation. The self-tuning regulator is a special case of linear quadratic control, and is computationally very simple. Aström and Wittenmark [4] described a self-tuning regulator for single input - single output systems which is based on minimum variance control. It does not penalize variations in the control action.

In this chapter, a self-tuning regulator for multivariable systems is described. It is based on minimum

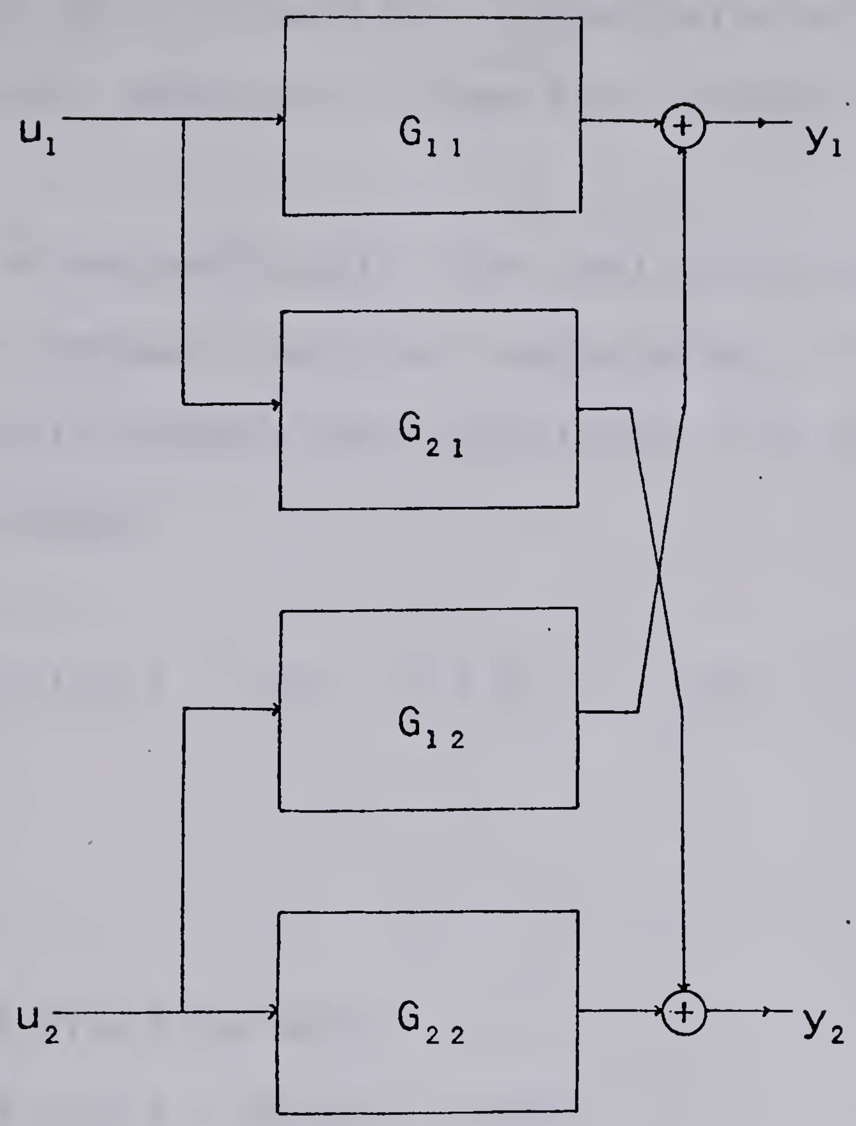


Figure 1: Transfer function model for an interactive system.

variance control and least squares esimation.

2.2 Minimum variance control for multivariable systems

Given a system with p measurable inputs and p measurable outputs and a time delay of k sampling intervals, the object of minimum variance control is to devise a control strategy which minimizes the following performance index:

$$\min E\{\underline{y}^T(t+k+1)\underline{\underline{Q}}\,\underline{y}(t+k+1)\} \tag{1}$$

where $\underline{\underline{Q}}$ is a positive semidefinite matrix

and the minimum is taken with respect to $\underline{u}(t)$

In order to be realizable, the control strategy must be a function of known inputs and outputs only. The proposed solution will satisfy this condition. Let the process model be described by:

$$\underline{\underline{A}}(q^{-1})\underline{y}(t) = \underline{\underline{B}}(q^{-1})\underline{u}(t-k-1) + \underline{\underline{C}}(q^{-1})\underline{e}(t) \qquad (2)$$

where

k is the time delay

$\underline{y}$ is a p x 1 output vector

$\underline{u}$ is a p x 1 input vector

$\{\underline{e}(t)\}$ is a sequence of white noise vectors with zero mean and covariance $\underline{\underline{R}} = E\{\underline{e}(t)\underline{e}^T(t)\}$

$\underline{\underline{A}}(q^{-1}) = \underline{\underline{I}} + \underline{\underline{A}}_1(q^{-1}) + \ldots + \underline{\underline{A}}_n(q^{-n})$ p x p matrix

$\underline{\underline{B}}(q^{-1}) = \underline{\underline{B}}_0 + \ldots + \underline{\underline{B}}_m(q^{-m})$ p x p matrix

$\underline{\underline{C}}(q^{-1}) = \underline{\underline{I}} + \underline{\underline{C}}_1(q^{-1}) + \ldots + \underline{\underline{C}}_n(q^{-n})$ p x p matrix

It is assumed that det $\underline{\underline{B}}(q^{-1})$ and det $\underline{\underline{C}}(q^{-1})$ have all their zeros outside the unit disc in the q-plane. The restriction on the $\underline{\underline{B}}$ matrix polynomial means that the system is minimum phase which is important to ensure that the closed loop system is stable although the minimum phase property is not relevant in the derivation of the control strategy. If

$\underline{\underline{B}}(q^{-1})$ has zeros inside the unit disc, the controller will try to cancel these zeros by corresponding poles. As a result the closed loop system will be very sensitive to parameter variations and this will cause problems in any practical application. The restriction on the $\underline{\underline{C}}$ matrix polynomial can be considered to be a weak condition. In order to obtain the optimal controller, new polynomial matrices are introduced that satisfy the following identity:

$$\underline{\underline{C}}(q^{-1}) = \underline{\underline{A}}(q^{-1})\underline{\underline{F}}(q^{-1}) + q^{-k-1}\underline{\underline{G}}(q^{-1}) \tag{3}$$

where $\underline{\underline{F}}$ and $\underline{\underline{G}}$ are polynomial matrices of degree k and n-1 respectively. Since $\underline{\underline{A}}(0) = \underline{\underline{I}}$ is nonsingular, $\underline{\underline{F}}$ and $\underline{\underline{G}}$ are unique [6].

At this point the analogy between the single input - single output and the multivariable case stops. The reason for this is that in general matrix multiplication is not commutative. In order to solve the problem it is necessary to define two other polynomial matrices $\tilde{\underline{\underline{F}}}(q^{-1})$ and $\tilde{\underline{\underline{G}}}(q^{-1})$ which satisfy the conditions:

$$\begin{aligned} &\tilde{\underline{\underline{F}}}(q^{-1})\underline{\underline{G}}(q^{-1}) = \tilde{\underline{\underline{G}}}(q^{-1})\underline{\underline{F}}(q^{-1}) \\ &\det \tilde{\underline{\underline{F}}}(q^{-1}) = \det \underline{\underline{F}}(q^{-1}) \\ &\tilde{\underline{\underline{F}}}(0) = \underline{\underline{F}}(0) = \underline{\underline{I}} \end{aligned} \tag{4}$$

and $\tilde{\underline{\underline{F}}}$ and $\tilde{\underline{\underline{G}}}$ always exist but are not unique. The polynomial matrix $\tilde{\underline{\underline{C}}}(q^{-1})$ defined by:

$$\tilde{\underline{\underline{C}}}(q^{-1}) = \tilde{\underline{\underline{F}}}(q^{-1})\underline{\underline{A}}(q^{-1}) + q^{-k-1}\tilde{\underline{\underline{G}}}(q^{-1}) \tag{5}$$

has the properties:

$$\tilde{\underline{\underline{C}}}(0) = \underline{\underline{I}}$$

$$\text{since } \tilde{\underline{\underline{F}}}(0) = \underline{\underline{A}}(0) = \underline{\underline{I}}$$

$$\tilde{\underline{\underline{F}}}(q^{-1})\underline{\underline{C}}(q^{-1}) = \tilde{\underline{\underline{C}}}(q^{-1})\underline{\underline{F}}(q^{-1}) \tag{6}$$

$$\det \underline{\underline{C}}(q^{-1}) = \det \tilde{\underline{\underline{C}}}(q^{-1})$$

Hence, this implies that det $\tilde{\underline{\underline{C}}}(q^{-1})$ has all zeros outside the unit disc. Rewriting the process model as:

$$\underline{\underline{A}}(q^{-1})\underline{y}(t+k+1) = \underline{\underline{B}}(q^{-1})\underline{u}(t) + \underline{\underline{C}}(q^{-1})\underline{e}(t+k+1) \tag{7}$$

the optimal control strategy can be derived. Equation (7), premultiplied by $\tilde{\underline{\underline{F}}}(q^{-1})$ gives:

$$\tilde{\underline{\underline{F}}}(q^{-1})\underline{\underline{A}}(q^{-1})\underline{y}(t+k+1)$$
$$= \tilde{\underline{\underline{F}}}(q^{-1})\underline{\underline{B}}(q^{-1})\underline{u}(t) + \tilde{\underline{\underline{F}}}(q^{-1})\underline{\underline{C}}(q^{-1})\underline{e}(t+k+1) \tag{8}$$

Substitution of equations (5) and (6) into equation (8) gives:

$$\tilde{\underline{\underline{C}}}(q^{-1})\{\underline{y}(t+k+1) - \underline{\underline{F}}(q^{-1})\underline{e}(t+k+1)\}$$
$$= \tilde{\underline{\underline{G}}}(q^{-1})\underline{y}(t) + \tilde{\underline{\underline{F}}}(q^{-1})\underline{\underline{B}}(q^{-1})\underline{u}(t) \tag{9}$$

Defining the function $\underline{w}(t)$ as:

$$\underline{w}(t) = \tilde{\underline{\underline{G}}}(q^{-1})\underline{y}(t) + \tilde{\underline{\underline{F}}}(q^{-1})\underline{\underline{B}}(q^{-1})\underline{u}(t) \tag{10}$$

The solution of equation (9) can be written as:

$$\begin{aligned}\underline{y}(t+k+1) &= \underline{\underline{F}}(q^{-1})\underline{e}(t+k+1)\\ &\quad + \Sigma\, \underline{\underline{M}}_{t-s}\underline{w}(s) + I(t,t_0)\end{aligned} \tag{11}$$

where Σ is taken for $s=t,t_0$

$\underline{\underline{M}}_{t-s}$ are p x p matrices

$I(t,t_0)$ is a vector function which depends on the initial conditions

The noise vectors $\underline{e}(t+k+1)$, $\underline{e}(t+k)$, ...$\underline{e}(t+1)$ are independent of $\underline{y}(t)$, $\underline{y}(t-1)$, ...$\underline{u}(t-1)$, $\underline{u}(t-2)$, ... and of the initial conditions. Hence

$$\begin{aligned}&E\{\underline{y}^T(t+k+1)\underline{\underline{Q}}\underline{y}(t+k+1)\}\geq\\ &\qquad E\{[\underline{\underline{F}}(q^{-1})\underline{e}(t+k+1)]^T\underline{\underline{Q}}\underline{\underline{F}}(q^{-1})\underline{e}(t+k+1)\}\end{aligned} \tag{12}$$

The equality holds when $I(t,t_0) = 0$ and $\underline{w}(s) = 0$
This gives the following control law:

$$\tilde{\underline{\underline{G}}}(q^{-1})\underline{y}(t) + \tilde{\underline{\underline{F}}}(q^{-1})\underline{\underline{B}}(q^{-1})\underline{u}(t) = 0 \tag{13}$$

The elements of $I(t,t_0)$ will go to zero exponentially when $\underline{w}(t) = 0$ because det $\tilde{\underline{\underline{C}}}(q^{-1})$ has all zeros outside the unit disc. If det $\tilde{\underline{\underline{C}}}(q^{-1})$ has zeros inside the unit disc, not all

elements of $I(t,t_0)$ go to zero exponentially and the optimal control is not obtained until these have been damped out. The control error for the system established in equation (2) with arbitrary initial conditions is asymptotically given by:

$$\underline{y}(t) = \underline{\underline{F}}(q^{-1})\underline{e}(t) \tag{14}$$

The control strategy in equation (13) is realizable. Unlike in single input - single output minimum variance control the control parameters are not unique. However, if $\underline{\underline{C}}(q^{-1}) = \underline{\underline{I}}$, the control strategy becomes:

$$\underline{\underline{B}}(q^{-1})\underline{u}(t) = [\underline{\underline{A}}(q^{-1}) - \underline{\underline{I}}]\underline{y}(t+k+1) \tag{15}$$

and the control error is:

$$\underline{y}(t) = \underline{e}(t) \tag{16}$$

In the previous analysis, the set point is assumed to be zero, or did not show up explicitly in the equations because the system was normalized. Explicit incorporation of the set point is done by changing the performance index to:

$$\min E\{[\underline{y}(t+k+1) - \underline{y}_s]^T \underline{\underline{Q}}[\underline{y}(t+k+1) - \underline{y}_s]\} \tag{17}$$

where $\underline{y}_s$ is the set point vector. The control function is then changed to:

$$\tilde{\underline{\underline{G}}}(q^{-1})[\underline{y}(t) - \underline{y}_s] + \tilde{\underline{\underline{F}}}(q^{-1})\underline{\underline{B}}(q^{-1})\underline{u}(t) = 0 \tag{18}$$

The minimum variance controller is restricted to minimum phase systems, and the strategy does not penalize the control variable. In spite of this it is an interesting application because of its computational simplicity. It has been used succesfully to control both single input - single output and multivariable systems.

2.3 Least squares estimation

The multivariable self-tuning regulator presented in this thesis is based on a least squares estimator and a linear controller established at every sampling instant using the current estimates. Least squares estimation is a method of identifying parameters of a system given the mathematical structure or description of the process. It is assumed that the system can be represented by a set of linear equations in terms of a finite number of past inputs and outputs. Although most processes are nonlinear, it is often possible to find a linear description which is good for small perturbations around a steady state. Let the process be described by:

$$\begin{aligned}[\underline{\underline{I}} + \underline{\underline{A}}_1 q^{-1} + \ldots + \underline{\underline{A}}_n q^{-n}]\underline{y}(t) = \\ [\underline{\underline{B}}_0 + \ldots + \underline{\underline{B}}_m q^{-m}]\underline{u}(t-1)\end{aligned} \tag{19}$$

$\underline{\underline{A}}_i$, $\underline{\underline{B}}_i$ are p x p matrices

$\underline{y}$, $\underline{u}$ are 1 x p vectors

The object is to estimate the parameter matrices from measurements of inputs u_1, ...u_p and outputs y_1,...y_p. Let n≥m; to estimate the parameters at sampling instant (n+N), the output vectors

$\underline{y}(0)$, $\underline{y}(1)$, ... $\underline{y}(n+N)$

and the input vectors

$\underline{u}(n-m)$, ... $\underline{u}(n+N-1)$

are required. If m>n a larger set of measurements is needed, however the method is essentially the same.

At any time t, the prediction error can be defined as the difference between the measured output and the prediction of the output, based on previous measurements and an estimate of the parameters.

$$\underline{e}(t) = \underline{y}(t) - \underline{\underline{\theta}}\underline{\psi}(t) \tag{20}$$

$$\underline{\psi}^T(t) = [\underline{u}^T(t-1) \ \ldots \ \underline{u}^T(t-m) \ \underline{y}^T(t-1) \ \ldots \ \underline{y}^T(t-n)] \tag{21}$$

$$\underline{\underline{\theta}} = [\underline{\underline{\beta}}0 \ \ldots \ \underline{\underline{\beta}}m \ -\underline{\underline{\alpha}}1 \ \ldots \ -\underline{\underline{\alpha}}n] \tag{22}$$

$\underline{\underline{\beta}}i$ is an estimate for $\underline{\underline{B}}_i$

$\underline{\underline{\alpha}}i$ is an estimate for $\underline{\underline{A}}_i$

Having a set of n+N measurements of output and input a matrix equation of errors can be written:

$$\underline{e}^T(n) = \underline{y}^T(n) - [\underline{u}^T(n-1) \ \ldots \ \underline{u}^T(n-m) \ \underline{y}^T(n-1) \ \ldots \ \underline{y}^T(0)\ ] \ \underline{\underline{\theta}}^T$$

$$\underline{e}^T(n+1) = \underline{y}^T(n+1) -$$
$$[\underline{u}^T(n) \;\ldots\; \underline{u}^T(n-m+1) \;\; \underline{y}^T(n) \;\ldots\; \underline{y}^T(1)\;] \;\underline{\underline{\theta}}^T$$

...

...

$$\underline{e}^T(n+N) = \underline{y}^T(n+N) -$$
$$[\underline{u}^T(n+N-1)\ldots\underline{u}^T(n+N-m) \;\; \underline{y}^T(n+N-1)\ldots\underline{y}^T(N)] \;\underline{\underline{\theta}}^T$$

or in matrix form:

$$\underline{\bar{e}}^T = \underline{\bar{y}}^T - \underline{\underline{\Psi}}^T \underline{\underline{\theta}}^T \tag{23}$$

Minimizing the error matrix

$$\underline{\underline{V}} = \underline{\bar{e}}^T \underline{\bar{e}} \tag{24}$$

gives an estimate for the parameter matrix

$$\hat{\underline{\underline{\theta}}}^T = (\underline{\underline{\Psi}}\underline{\underline{\Psi}}^T)^{-1}\underline{\underline{\Psi}}\underline{\bar{y}} \tag{25}$$

It is useful to write this equation in a recursive form, so that it can be used for on-line identification. This will also allow the algorithm to follow parameter changes. Defining:

$$n+N = t$$
$$\underline{\underline{P}}(t) = [\underline{\underline{\Psi}}(t)\underline{\underline{\Psi}}^T(t)]^{-1} \tag{26}$$

$$\bar{\underline{y}}(t+1) = \begin{vmatrix} \bar{\underline{y}}(t) \\ \underline{y}^T(t+1) \end{vmatrix} \tag{27}$$

$$\underline{\underline{\Psi}}^T(t+1) = \begin{vmatrix} \underline{\underline{\Psi}}^T(t) \\ \underline{\psi}^T(t+1) \end{vmatrix} \tag{28}$$

allows $\underline{\underline{P}}(t+1)$ to be expressed as:

$$\begin{aligned} \underline{\underline{P}}(t+1) &= [\underline{\underline{\Psi}}(t)\underline{\underline{\Psi}}^T(t) + \underline{\psi}(t+1)\underline{\psi}^T(t+1)]^{-1} \\ &= [\underline{\underline{P}}^{-1}(t) + \underline{\psi}(t+1)\underline{\psi}^T(t+1)]^{-1} \end{aligned} \tag{29}$$

and $\hat{\underline{\underline{\theta}}}^T(t+1)$ as:

$$\hat{\underline{\underline{\theta}}}^T(t+1) = \underline{\underline{P}}(t+1) \begin{vmatrix} \underline{\underline{\Psi}}^T(t) \\ \underline{\psi}^T(t+1) \end{vmatrix}^T \begin{vmatrix} \bar{\underline{y}}(t) \\ \underline{y}^T(t+1) \end{vmatrix} \tag{30}$$

$$= \underline{\underline{P}}(t+1)\{\underline{\underline{\Psi}}(t)\bar{\underline{y}}(t) + \underline{\psi}(t+1)\underline{y}^T(t+1)\}$$

$$\begin{aligned} \hat{\underline{\underline{\theta}}}^T(t+1) &= \underline{\underline{P}}(t+1)\ \{\underline{\underline{P}}^{-1}(t)\hat{\underline{\underline{\theta}}}^T(t) + \underline{\psi}(t+1)\underline{y}^T(t+1)\} \\ &= \hat{\underline{\underline{\theta}}}^T(t) + \{[\underline{\underline{P}}(t+1)\underline{\underline{P}}^{-1}(t) - \underline{\underline{I}}]\hat{\underline{\underline{\theta}}}^T(t) + \\ &\quad \underline{\underline{P}}(t+1)\underline{\psi}(t+1)\underline{y}^T(t+1)\} \\ &= \hat{\underline{\underline{\theta}}}^T(t) + \\ &\quad \underline{\underline{P}}(t+1)\underline{\psi}(t+1)[\underline{y}^T(t+1) - \underline{\psi}^T(t+1)\hat{\underline{\underline{\theta}}}^T(t)] \end{aligned} \tag{31}$$

Thus, the new estimate of the parameters is the old estimate plus the prediction error multiplied by a gain factor. Equation (31) is in a standard recursive form. Equation (29) can be written in a form which is easier to compute because

no matrix inversion is required. This can be done as follows:

$$\underline{\underline{P}}^{-1}(t+1) = \underline{\underline{P}}^{-1}(t) + \underline{\psi}(t+1)\underline{\psi}^{T}(t+1)$$

$$\underline{\underline{P}}^{-1}(t+1)\underline{\underline{P}}(t)\underline{\psi}(t+1) = \underline{\psi}(t+1) + \underline{\psi}(t+1)\underline{\psi}^{T}(t+1)\underline{\underline{P}}(t)\underline{\psi}(t+1)$$

$$\underline{\underline{P}}^{-1}(t+1)\underline{\underline{P}}(t)\underline{\psi}(t+1) = \underline{\psi}(t+1)[1 + \underline{\psi}^{T}(t+1)\underline{\underline{P}}(t)\underline{\psi}(t+1)]$$

$$\underline{\underline{P}}^{-1}(t+1)\underline{\underline{P}}(t)\underline{\psi}(t+1)[1 + \underline{\psi}^{T}(t+1)\underline{\underline{P}}(t)\underline{\psi}(t+1)]^{-1} = \underline{\psi}(t+1)$$

$$\begin{aligned}\underline{\underline{P}}^{-1}(t+1)\underline{\underline{P}}(t)\underline{\psi}(t+1)[1 + \underline{\psi}^{T}(t+1)\underline{\underline{P}}(t)\underline{\psi}(t+1)]^{-1} & \\ \underline{\psi}^{T}(t+1)\underline{\underline{P}}(t) & \\ = \underline{\psi}(t+1)\underline{\psi}^{T}(t+1)\underline{\underline{P}}(t) & \\ = \underline{\underline{P}}^{-1}(t+1)\underline{\underline{P}}(t) - \underline{\underline{I}} &\end{aligned}$$

$$\underline{\underline{P}}(t)\underline{\psi}(t+1)[1 + \underline{\psi}^{T}(t+1)\underline{\underline{P}}(t)\underline{\psi}(t+1)]^{-1}\underline{\psi}^{T}(t+1)\underline{\underline{P}}(t) = \underline{\underline{P}}(t) - \underline{\underline{P}}(t+1)$$

$$\underline{\underline{P}}(t+1) = \underline{\underline{P}}(t) - \underline{\underline{P}}(t)\underline{\psi}(t+1)[1 + \underline{\psi}^{T}(t+1)\underline{\underline{P}}(t)\underline{\psi}(t+1)]^{-1}\underline{\psi}^{T}(t+1)\underline{\underline{P}}(t)$$

The gain factor in the estimation equation is usually called $\underline{\underline{K}}(t)$

$$\underline{\underline{K}}(t) = \underline{\underline{P}}(t+1)\underline{\psi}(t+1)$$

$$\underline{\underline{K}}(t) = \underline{\underline{P}}(t)\underline{\psi}(t+1) \{1 - [1 + \underline{\psi}^T(t+1)\underline{\underline{P}}(t)\underline{\psi}(t+1)]^{-1}\underline{\psi}^T(t+1)\underline{\underline{P}}(t)\underline{\psi}(t+1)\}$$

$$\underline{\underline{K}}(t) = \underline{\underline{P}}(t)\underline{\psi}(t+1)[1 + \underline{\psi}^T(t+1)\underline{\underline{P}}(t)\underline{\psi}(t+1)]^{-1}$$

Hence, the complete recursive algorithm is:

$$\hat{\underline{\underline{\theta}}}^T(t+1) = \hat{\underline{\underline{\theta}}}^T(t) + \underline{\underline{K}}(t)[\underline{y}^T(t+1) - \underline{\psi}^T(t+1)\hat{\underline{\underline{\theta}}}^T(t)] \quad (32)$$

$$\underline{\underline{K}}(t) = \underline{\underline{P}}(t)\underline{\psi}(t+1)[1 + \underline{\psi}^T(t+1)\underline{\underline{P}}(t)\underline{\psi}(t+1)]^{-1} \quad (33)$$

$$\underline{\underline{P}}(t+1) = \underline{\underline{P}}(t) - \underline{\underline{K}}(t)\underline{\psi}^T(t+1)\underline{\underline{P}}(t) \quad (34)$$

To start the estimation algorithm one needs:

n+1 measurements of the output signals

m measurements of the input signals

an initial estimate for $\underline{\underline{\theta}}$

an initial estimate for $\underline{\underline{P}}$

Each recursive step involves:

(a) measuring inputs and outputs

(b) forming $\underline{\psi}(t+1)$

(c) solving equations (32), (33) and (34) in that order

It is evident that the output signals must contain enough information about the system for the method to work. If $\underline{y}$ and $\underline{u}$ do not change in time, the matrix $\underline{\underline{\Psi}}$ will lose rank and $(\underline{\underline{\Psi}}\underline{\underline{\Psi}}^T)$ becomes singular. Therefore, equation (26) is not defined.

For systems with time delays a similar estimation algorithm can be derived.

2.4 Self-tuning controllers

The design of a controller for a multivariable system is usually based on an extensive knowledge of the process. However, in practical applications the process dynamics are often unknown and a model must be obtained by using identification techniques, before the controller can be designed. Self-tuning controllers simplify this procedure because only some parameters of the model must be determined before the algorithm can be applied to the system. The method of self-tuning control involves two basic steps for each sampling interval. First the parameters of a prediction model are estimated and then a control law based on the current estimates is derived. In this chapter, a method which was originally proposed by Aström [4] for single input - single output systems, is described for multivariable systems. The proposed algorithm is a multivariable control scheme based on least squares estimation and minimum variance control.

Assume a process that can be represented by a linear system with p inputs, p outputs and a time delay equal to k sample intervals:

$$[\underline{\underline{I}} + \underline{\underline{A}}_1 q^{-1} + \ldots + \underline{\underline{A}}_n q^{-n}]\underline{y}(t) = \qquad (35)$$
$$[\underline{\underline{B}}_0 q^{-k} + \ldots + \underline{\underline{B}}_m q^{-k-m}]\underline{u}(t-1) +$$
$$[\underline{\underline{I}} + \underline{\underline{C}}_1 q^{-1} + \ldots + \underline{\underline{C}}_n q^{-n}]\underline{e}(t)$$

where $\underline{e}(t)$ is a white noise sequence with zero mean

Because the least squares estimator does not estimate the $\underline{\underline{C}}$ parameter matrices, this model can not be used in the algorithm. A prediction model of the following form allows the algorithm to estimate the parameters recursively and to predict the output signals k steps ahead:

$$[\underline{\underline{I}} + \underline{\underline{A}}1\ q^{-k-1} + \ldots + \underline{\underline{A}}n\ q^{-k-n}]\underline{y}(t) = \qquad (36)$$
$$[\underline{\underline{B}}0\ q^{-k-1} + \ldots + \underline{\underline{B}}m{+}k\ q^{-2k-m-1}]\underline{u}(t) + \underline{\varepsilon}(t)$$

where $\underline{\varepsilon}(t)$ is supposed to be white noise with zero mean

By choosing this model structure for the identification, the parameters of the controller are obtained from the estimator without further computations. If the objective is to make the vector of outputs equal to a vector of reference values $\underline{y}_r$, then in the case of minimum variance control the

self-tuning algorithm becomes:

$$\underline{\underline{\hat{\theta}}}^T(t+1) = \underline{\underline{\hat{\theta}}}^T(t) + \underline{\underline{K}}(t)[\underline{y}^T(t+1) - \underline{\psi}^T(t-k+1)\underline{\underline{\hat{\theta}}}^T(t)] \quad (37)$$

$$\underline{\underline{K}}(t) = \underline{\underline{P}}(t)\underline{\psi}(t-k+1)[1 + \underline{\psi}^T(t-k+1)\underline{\underline{P}}(t)\underline{\psi}(t-k+1)]^{-1} \quad (38)$$

$$\underline{\underline{P}}(t+1) = \underline{\underline{P}}(t) - \underline{\underline{K}}(t)\underline{\psi}^T(t-k+1)\underline{\underline{P}}(t) \quad (39)$$

$$\begin{aligned}\underline{u}(t-k) = {} & \underline{\underline{\hat{\beta}}}_0^{-1}\{\underline{y}_r - \underline{\underline{\hat{\beta}}}_1\underline{u}(t-k-1) - \ldots - \underline{\underline{\hat{\beta}}}_{m+k}\underline{u}(t-2k-m) \\ & + \underline{\underline{\hat{\alpha}}}_1\underline{y}(t-k) + \ldots + \underline{\underline{\hat{\alpha}}}_n\underline{y}(t-k-n+1)\} \end{aligned} \quad (40)$$

Convergence studies of this type of self-tuning regulator have been done by Ljung [15] for the single input - single output case and by Borisson [6] for the multivariable case. It has been shown that convergence depends upon three conditions: a noise condition, input and output boundedness, and stability. The noise condition can be expressed as a linear stochastic problem and the stability condition concerns the stability of a set of nonlinear ordinary differential equations. The solutions of these equations will approximate the trajectories of the estimates [6]. Ljung and Wittenmark [15] proved that the single input - single output STR algorithm does not always converge. However under certain conditions the self-tuning regulator will stabilize a minimum phase system, even if the estimated parameters do not converge.

Convergence analysis in the multivariable case requires

the solution of a set of very complex nonlinear differential equations. Only the results of the analysis of the asymptotic behaviour of the regulator, as given by Borisson, will be repeated here.

Let $\{\underline{y}(t)\}$ and $\{\underline{u}(t)\}$ be uniformly bounded
If $\underline{y}$ converges to $\underline{y}_o$
and $\underline{u}$ converges to $\underline{u}_o$
and the estimated parameters converge as N tends to infinity, then:

$$\lim_{N\to\infty} 1/N \sum_{t=1}^{N} \underline{\psi}(t-k-1)\underline{\psi}^T(t-k-1)[\hat{\underline{\underline{\theta}}}^T(t-k-1) - \hat{\underline{\underline{\theta}}}^T(N)] = \underline{0}$$

$$\lim_{N\to\infty} 1/N \sum_{t=1}^{N} \underline{y}_o(t+\tau)\underline{y}_o^T(t) = \lim_{N\to\infty} 1/N \sum_{t=1}^{N} \underline{y}(t+\tau)\underline{y}^T(t) = \underline{\underline{0}}$$
$$\tau = k+1,\ldots k+n+1$$

$$\lim_{N\to\infty} 1/N \sum_{t=1}^{N} \underline{y}_o(t+\tau)\underline{u}_o^T(t) = \lim_{N\to\infty} 1/N \sum_{t=1}^{N} \underline{y}(t+\tau)\underline{u}^T(t) = \underline{\underline{0}}$$
$$\tau = k+1,\ldots k+m+1$$

Aström and Wittenmark [3] proved that the single input - single output self-tuning regulator always gives minimum variance control for a general noise filter C, if the estimated parameters converge in such a way that the resulting α and β polynomials are relatively prime and if the structure of the model is correct.

Borisson proves that this result also holds for multivariable systems if $\underline{\underline{C}}(q^{-1}) = \underline{\underline{I}}$. Under additional

conditions the result also applies to a class of multivariable systems with a general $\underline{\underline{C}}$-polynomial matrix [6].

The self-tuning controller algorithm as described here is basically the same as that proposed by Borisson, and by Keviczky et al [11]. Keviczky uses a different notation and does not give a separate equation for the gain factor. This allows for compact programming. Borisson divides the estimation procedure into p steps; in every step the parameters corresponding to one output variable are estimated. Equation (37) becomes:

$$\underline{\hat{\theta}}_i(t+1) = \underline{\hat{\theta}}_i(t) + \underline{K}(t)[y_i(t+1) - \underline{\psi}^T(t-k)\underline{\hat{\theta}}_i(t)] \qquad (41)$$

with $\underline{\underline{\theta}}^T = [\underline{\theta}_1 \ \ldots \ \underline{\theta}_p]$

and $\underline{y}^T = [y_1 \ \ldots \ y_p]$

This procedure reduces the amount of memory needed for execution of the algorithm provided that the same $\underline{\underline{P}}(0)$ is used for every output variable. It follows that, using the transformation:

$$\underline{u}'(t) = B0\underline{u}(t)$$

so that $\underline{\underline{\beta}}0$ is a unit matrix, the algorithm can be considered to be an interconnection of p single input - single output self-tuning regulators with all the interacting signals treated as "feedforward" signals.

3. MODIFICATIONS OF THE BASIC ALGORITHM

3.1 Introduction

Simulation results of systems controlled by multivariable self-tuning controllers show that the algorithm works well for the class of systems for which it is designed i.e. minimum phase systems. For certain systems it is possible to improve the process output by making some modifications to the basic controller. Exponential forgetting factors were added to the single input - single output self-tuning regulator by Aström and Wittenmark [4]. They are equally useful in the multivariable case.

Fixing one parameter, usually the leading β parameter, in the estimation algorithm has almost been standard procedure since the first introduction of self-tuning regulators, although it is possible to estimate all parameters. In the single input - single output case the choice of this parameter is less crucial than for multivariable systems. Simulations showed that in the latter case it is very difficult to fix one parameter matrix unless a very good estimate of its actual value is available.

Integral control action and feedforward compensation are procedures that are often added to basic control schemes in order to improve the output performance when operating the process under certain conditions. Simulations proved that they were also valuable additions to the basic self-tuning controller.

In the last section of this chapter a method which can improve the average characteristics of a process output is described.

3.2 Exponential forgetting factors

If the gain factor $\underline{\underline{K}}(t)$ becomes very small, and the parameter estimates have not converged yet, then the change in the estimates at each sampling instant will be very small and convergence is slow. To prevent this, one can introduce exponential weighting of past data. This can be done by changing equation (34) to:

$$\underline{\underline{P}}(t+1) = 1/\omega^2[\underline{\underline{P}}(t) - \underline{\underline{K}}(t)\underline{\psi}(t+1)\underline{\underline{P}}(t)]$$

The weight on past data is ω^{2n} after n steps. This means that approximately $2.3/(1-\omega^2)$ values are remembered and that the weight has decreased to 0.1. Thus, old measurements of input and output signals which might be irrelevant to the new process situation are not used in the estimation algorithm.

Exponential forgetting factors are also useful in the case of disturbances. If $\underline{\underline{K}}(t)$ is small, the parameters will not be able to converge to their new values when a disturbance occurs. This will result in poor control. By choosing $\omega<1.0$, irrelevant past data will not be used in the algorithm. During steady-state operation (i.e. where no disturbances occur and the parameters have converged), a

small forgetting factor will cause fluctuations in the parameters and the output. In such a case it is better to set ω equal to one.

3.3 Estimation of the leading B matrix

In most applications of self-tuning regulators, one of the parameters, usually β_0, is given a fixed value. It is possible to include the estimation of B_0, or $\underline{\underline{B}}0$ in the multivariable case, in the algorithm provided that the estimate $\underline{\underline{\beta}}0$ is nonsingular at every instant because the control law requires $\underline{\underline{\beta}}0^{-1}$. A practical procedure to avoid the difficulty of singularity is to test $\det\underline{\underline{\beta}}0$ at every step, and to reset $\underline{\underline{\beta}}0$ to its previous value if the new matrix is singular.

If $\underline{\underline{\beta}}0$ is kept constant, it must be chosen in such a way that it does not prevent the estimated parameters from converging. The constraints are:

$$|\mu_i| < 1 \qquad\qquad i = 1, \ldots p$$

where μ_i are the eigenvalues of the matrix $[\underline{\underline{I}} - \underline{\underline{B}}0\underline{\underline{\beta}}0^{-1}]$, $\underline{\underline{B}}0$ is the true parameter matrix in the process model and $\underline{\underline{\beta}}0$ is the fixed value in the control algorithm. The proof of this theorem is given by Borisson [6]. In most cases however, $\underline{\underline{B}}0$ is not known, and it is difficult to find a value which fulfills the above condition. A possible way of solving this problem is to estimate $\underline{\underline{B}}0$ initally, and to keep it set to

the estimated value in subsequent runs.

The choice of $\underline{\underline{\beta}}0$ influences the output variance. Although convergence can be obtained for different values, the output variance is a minimum when $\underline{\underline{\beta}}0$ is kept constant at its true value $\underline{\underline{B}}0$.

3.4 Integral control

The self-tuning regulator as developed in Chapter 2 is not generally able to eliminate offset when a process is subjected to disturbances. For example if the feed flow rate to a binary distillation column changes, the basic self-tuning regulator (STR) is likely to produce an offset. If the disturbance is not measurable, the only way to eliminate this offset is by using integral control action. Similarly, in the case of set point changes, integral control action will prevent offset.
Rewriting the prediction algorithm as:

$$[\underline{\underline{I}} + \underline{\underline{A}}1\, q^{-k-1} + \ldots + \underline{\underline{A}}n{+}1\, q^{-k-n-1}]\underline{y}(t) \qquad (42)$$
$$= [\underline{\underline{B}}0\, q^{-k-1} + \ldots + \underline{\underline{B}}m\, q^{-k-m-1}][\underline{u}(t) - \underline{u}(t-1)] + \underline{\varepsilon}(t)$$

$$\underline{u}(t) - \underline{u}(t-1) = \Delta\underline{u}(t)$$

and $\underline{\psi}(t)$ as:

$$\underline{\psi}^T(t) = [\Delta\underline{u}^T(t-1) \ \dots \ \Delta\underline{u}^T(t-m)$$
$$\underline{y}^T(t-1) \ \dots \ \underline{y}^T(t-n-1)] \qquad (43)$$

the control action becomes:

$$\Delta\underline{u}(t-k) = \underline{\underline{\beta}}0^{-1}\{\underline{y}_r - \underline{\underline{\beta}}1\Delta\underline{u}(t-k-1) - \dots - \underline{\underline{\beta}}m\Delta\underline{u}(t-k-m)$$
$$+ \underline{\underline{\alpha}}1\underline{y}(t-k) + \dots + \underline{\underline{\alpha}}n+1\underline{y}(t-k-n)\} \qquad (44)$$

Note that one more parameter matrix must be estimated i.e. p^2 more parameters.

3.5 Feedforward control

If the disturbance which enters a process is measurable, it is possible to estimate the parameters of a disturbance model and to use them in feedforward control. The self-tuning regulator is especially suited for this technique because it is capable of estimating slowly varying parameters of a disturbance model. The prediction model now becomes:

$$\underline{\underline{A}}(q^{-1})\underline{y}(t) = \underline{\underline{B}}(q^{-1})\underline{u}(t) + \underline{\underline{G}}(q^{-1})\underline{z}(t) + \underline{\varepsilon}(t) \qquad (45)$$

where $\underline{z}$ is a vector of measurable disturbances.
The estimation algorithm remains the same but $\underline{\psi}(t)$ and $\underline{\underline{\theta}}$ are defined as:

$$\underline{\psi}^T(t) = [\underline{u}^T(t-1)\ldots\underline{u}^T(t-m)\ \underline{y}^T(t-1)\ldots\underline{y}^T(t-n)$$
$$\underline{z}^T(t-1)\ldots\underline{z}^T(t-f)] \quad (46)$$

$$\underline{\underline{\theta}} = [\underline{\underline{\beta}}0\ \underline{\underline{\beta}}2\ldots\underline{\underline{\beta}}m\ -\underline{\underline{\alpha}}1\ldots-\underline{\underline{\alpha}}n\ \underline{\underline{\gamma}}1\ldots\underline{\underline{\gamma}}f] \quad (47)$$

The controller includes feedforward action:

$$\underline{u}(t) = \underline{\underline{\beta}}0^{-1}\{\underline{y}_r - \underline{\underline{\beta}}1\underline{u}(t-1) - \ldots - \underline{\underline{\beta}}m\underline{u}(t-m) +$$
$$\underline{\underline{\alpha}}1(t) + \ldots + \underline{\underline{\alpha}}n\underline{y}(t-n+1) - \underline{\underline{\gamma}}1\underline{z}(t) - \ldots -$$
$$\underline{\underline{\gamma}}f\underline{z}(t-f+1) \quad (48)$$

Time delays in the process and in the disturbance can be included in the algorithm. Feedforward control and integral control can be combined when measurable and unmeasurable disturbances enter the process.

3.6 Output average over finite time

In many processes the product is blended in a tank or a process vessel, and in this case the average characteristics of the product are usually of more interest than the value of the output variable at all times. For example, if the liquid product from a process is flowing into a vessel, the resulting temperature and composition will be the average over the time period that the stream is flowing. If the instantaneous average is measured or estimated, and used as a feedback control signal, the average property in the vessel will be made equal to the specifications. If,

however, only instantaneous properties of the entering liquid are used as feedback signals, the average in the container may be different from the set point because there will be no compensation for the initial errors.

One way of making this offset zero is to change the set point so as to compensate for previous errors. The weighted average for a finite number of sample intervals N is:

$$y_a(N) = \sum_{i=1}^{N} h_i y(i) \Big/ \sum_{i=1}^{N} h_i \tag{49}$$

where h_i is the quantity of the process output over interval i.

The average of the process output should be as close as possible to the reference value y_r. This is achieved by the control law compensating the deviations of $y_a(K)$ from y_r for all $K \leq N$. In order to do this a prediction of $y_a(N)$ at every time instant K is needed. For the case where all h_i are equal the best prediction of $y_a(N)$ is given by:

$$\hat{y}_a(N|K) = [Ky_a(K) + (N-K)\hat{y}(K+1|K)]/N \tag{50}$$

$\hat{y}(K+1|K)$ is the one step ahead prediction of y

If there is a time delay k, the k+1 step ahead prediction of y_a is:

$$\hat{y}_a(K+k|K) = Ky_a(K) + y_{rv}(K+1) + \dots y_{rv}(K+k)]/K+k \tag{51}$$

and equation (50) becomes:

$$\hat{y}_a(N|K) = [(K+k)\hat{y}_a(K+k|K) + (N-K-k)\hat{y}(K+k+1|K)]/N \quad (52)$$

Since $y_a(N|K)$ should be equal to y_r, it is required that:

$$\hat{y}(K+k+1|K) = [Ny_r - (K+k)\hat{y}_a(K+k|K)]/(N-K-K) \quad (53)$$

and this value can be considered to be the new reference value $y_{rv}(K+k+1)$. In the multivariable case the control law is then changed to:

$$\underline{u}(t-k) = \underline{\underline{\beta}}0^{-1}[\underline{y}_{rm} - \underline{\underline{\beta}}1\underline{u}(t-k-1) \ \ldots + \underline{\underline{\alpha}}1\underline{y}(t-k) + \ \ldots]$$

$$\underline{y}_{rm} = \underline{y}_{rv}(K+k+1) \text{ for } K = 1,2 \ldots N-k-1$$
$$\underline{y}_{rm} = \underline{y}_r \text{ for } K = N-k, \ldots N$$

This technique was first described by Keviczky et al [11]. Their simulation results show that in certain cases varying the reference value decreases the loss:

$$E\{[y_a(N) - y_r]^T[y_a(N) - y_r]\}$$

4. SIMULATION OF THE SELF-TUNING CONTROL OF MULTIVARIABLE LINEAR SYSTEMS

4.1 Introduction

The multivariable self-tuning controller described in Chapter 2 was tested by simulation using a variety of different linear systems to investigate its control performance. Three representative examples are described here. The initial values of the regulator parameters were always set to zero except for $\underline{\underline{\beta}}0$ which, if estimated, was initially equal to the identity matrix. This is the best choice if nothing is known about the actual process parameters. However, in practical applications a smoother start-up of the process is obtained if the parameters are initialized with estimates obtained from previous runs. The results of applying the modifications described in the previous chapter, are also analyzed.

A description and listings of the computer programs used for simulation of the linear systems and the self-tuning controllers on the Amdahl 470V/7 computer system at the University of Alberta, is given in Appendix A.

4.2 Minimum phase system without time delay

The behaviour of the controller when applied to a minimum phase system was studied using the second order system given by Keviczky et al [11]. The system representation is:

$$\underline{y}(t) = \underline{\underline{A}}1\ \underline{y}(t-1) + \underline{\underline{A}}2\ \underline{y}(t-2) + \underline{\underline{B}}0\ \underline{u}(t-1) + \underline{\underline{B}}1\ \underline{u}(t-2)$$
$$+ \underline{\underline{I}}\ \underline{e}(t) + \underline{\underline{C}}1\ \underline{e}(t-1) + \underline{\underline{C}}2\ \underline{e}(t-2) \qquad (54)$$

where

$$\underline{\underline{A}}1 = \begin{vmatrix} 1.5 & -0.3 \\ -0.2 & 1.5 \end{vmatrix} \qquad \underline{\underline{A}}2 = \begin{vmatrix} -0.54 & 0.1 \\ -0.1 & -0.56 \end{vmatrix}$$

$$\underline{\underline{B}}0 = \begin{vmatrix} 2.0 & -0.3 \\ 0.1 & 1.0 \end{vmatrix} \qquad \underline{\underline{B}}1 = \begin{vmatrix} -1.8 & 0.2 \\ -0.2 & -0.5 \end{vmatrix}$$

$$\underline{\underline{C}}1 = \begin{vmatrix} 0.2 & 0.1 \\ -0.1 & 0.2 \end{vmatrix} \qquad \underline{\underline{C}}2 = \begin{vmatrix} -0.48 & -0.2 \\ 0.2 & -0.24 \end{vmatrix}$$

If $\underline{e}(t)=0.0$, the minimum variance controller is:

$$\underline{u}(t) = \underline{\underline{B}}0^{-1}\{ -\underline{\underline{B}}1\underline{u}(t-1) \qquad (55)$$
$$- \underline{\underline{A}}1[\underline{y}(t-1)-\underline{y}_s] - \underline{\underline{A}}2[\underline{y}(t-2)-\underline{y}_s]\}$$

but if $\underline{e}(t)$ is considered to have a covariance matrix $\sigma\underline{\underline{I}}$, the minimum variance controller is defined by:

$$\tilde{\underline{\underline{G}}}(q^{-1})[\underline{y}(t) - \underline{y}_s] + \tilde{\underline{\underline{F}}}(q^{-1})\underline{\underline{B}}(q^{-1})\underline{u}(t) = \underline{0} \qquad (56)$$

where $\tilde{\underline{\underline{G}}}(q^{-1})$ and $\tilde{\underline{\underline{F}}}(q^{-1})$ are obtained by solving the following equations:

$$\underline{\underline{I}} + \underline{\underline{C}}1q^{-1} + \underline{\underline{C}}2q^{-1} = [\underline{\underline{I}} - \underline{\underline{A}}1q^{-1} - \underline{\underline{A}}2q^{-2}]\ \underline{\underline{F}}1 \\ + \underline{\underline{G}}1q^{-1} + \underline{\underline{G}}2q^{-2} \tag{57}$$

$$\underline{\underline{F}}1 = \tilde{\underline{\underline{F}}}1 = \underline{\underline{I}} \tag{58}$$

$$\tilde{\underline{\underline{F}}}1[\underline{\underline{G}}1 + \underline{\underline{G}}2q^{-1}] = [\tilde{\underline{\underline{G}}}1 + \tilde{\underline{\underline{G}}}2q^{-1}]\underline{\underline{F}}1 \tag{59}$$

with the solution given by:

$$\tilde{\underline{\underline{F}}}1 = \underline{\underline{I}}$$

$$\tilde{\underline{\underline{G}}}1 = \begin{vmatrix} 1.7 & -0.2 \\ -0.3 & 1.7 \end{vmatrix} \qquad \tilde{\underline{\underline{G}}}2 = \begin{vmatrix} -1.02 & -0.1 \\ 0.1 & -0.8 \end{vmatrix}$$

The behaviour of the output signals of this system under self-tuning control is shown in Figure 2a. The covariance of the noise $\underline{e}(t)$ is $0.01\underline{\underline{I}}$ and the initial estimates of the parameters in the control algorithm are set to zero, except for $\underline{\underline{\beta}}0$ which has to be nonsingular at all times and is initially chosen as a unit matrix. Figure 2b shows the behaviour of the input signals and Figures 2c and 2d the parameter estimates. All parameters converge to those corresponding to the minimum variance controller but the β-parameters take much longer than the α-parameters. This does not prevent very good control. Kevizcky et al obtained similar results for the behaviour of the output signals, using a different value for the covariance of the noise.

Figures 3a and 3b show the behaviour of the output and

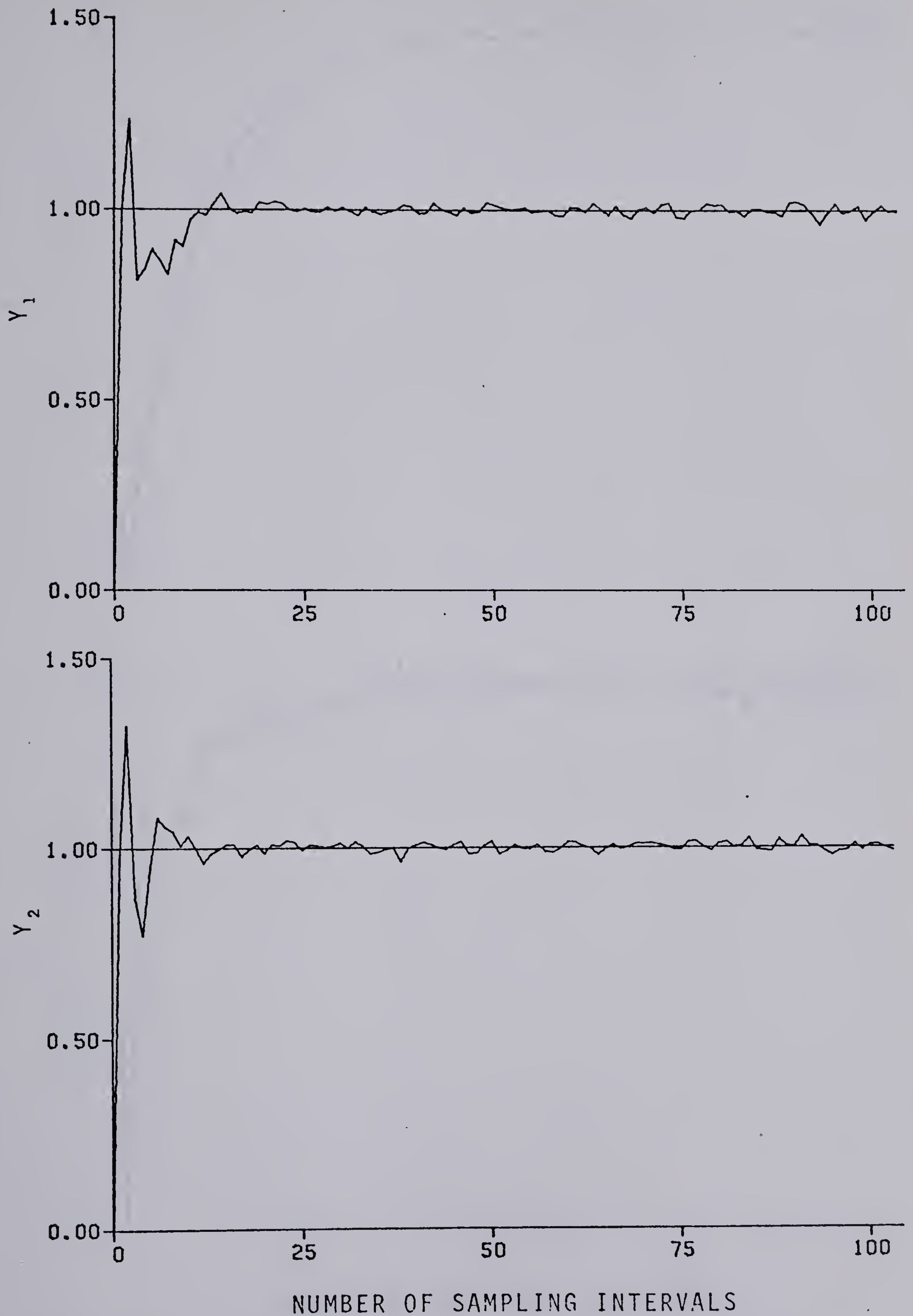


Figure 2a: Simulation of Keviczky second order system. Output behaviour when $\underline{e}(t)=0.01I$.

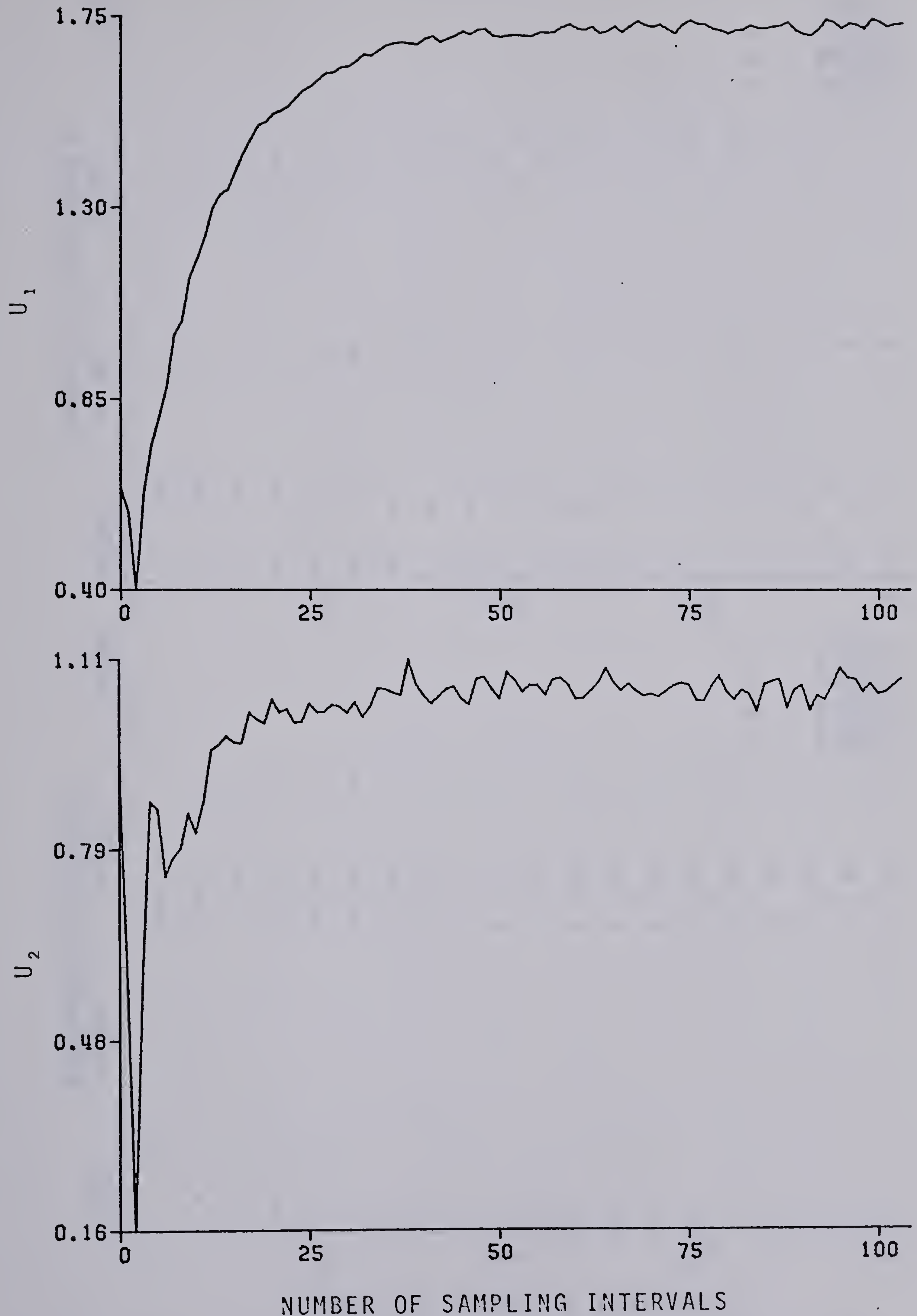


Figure 2b: Simulation of Keviczky second order system. Input behaviour when $\underline{e}(t)=0.01I$.

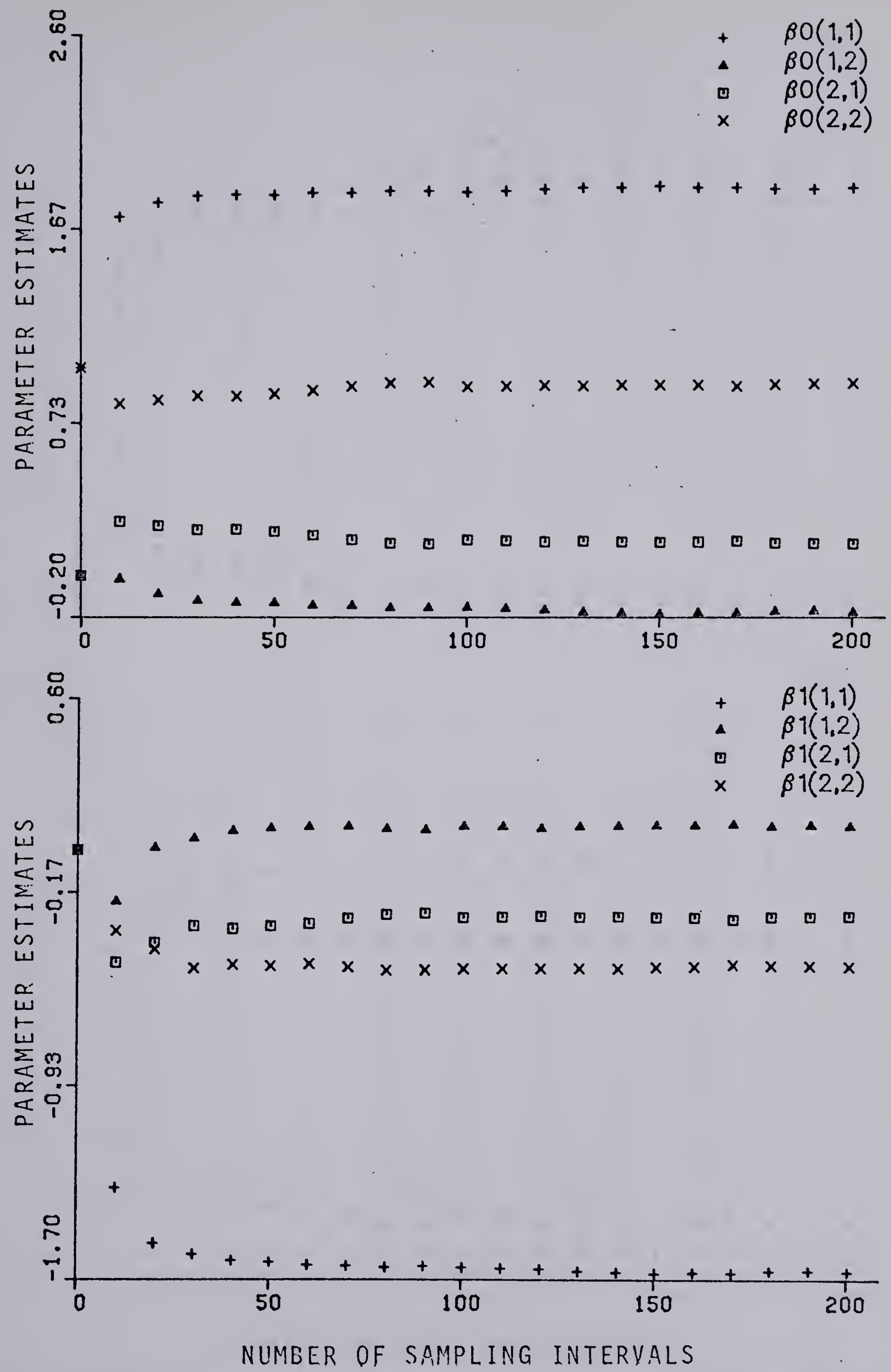


Figure 2c: Simulation of Keviczky second order system. Behaviour of the β parameter estimates.

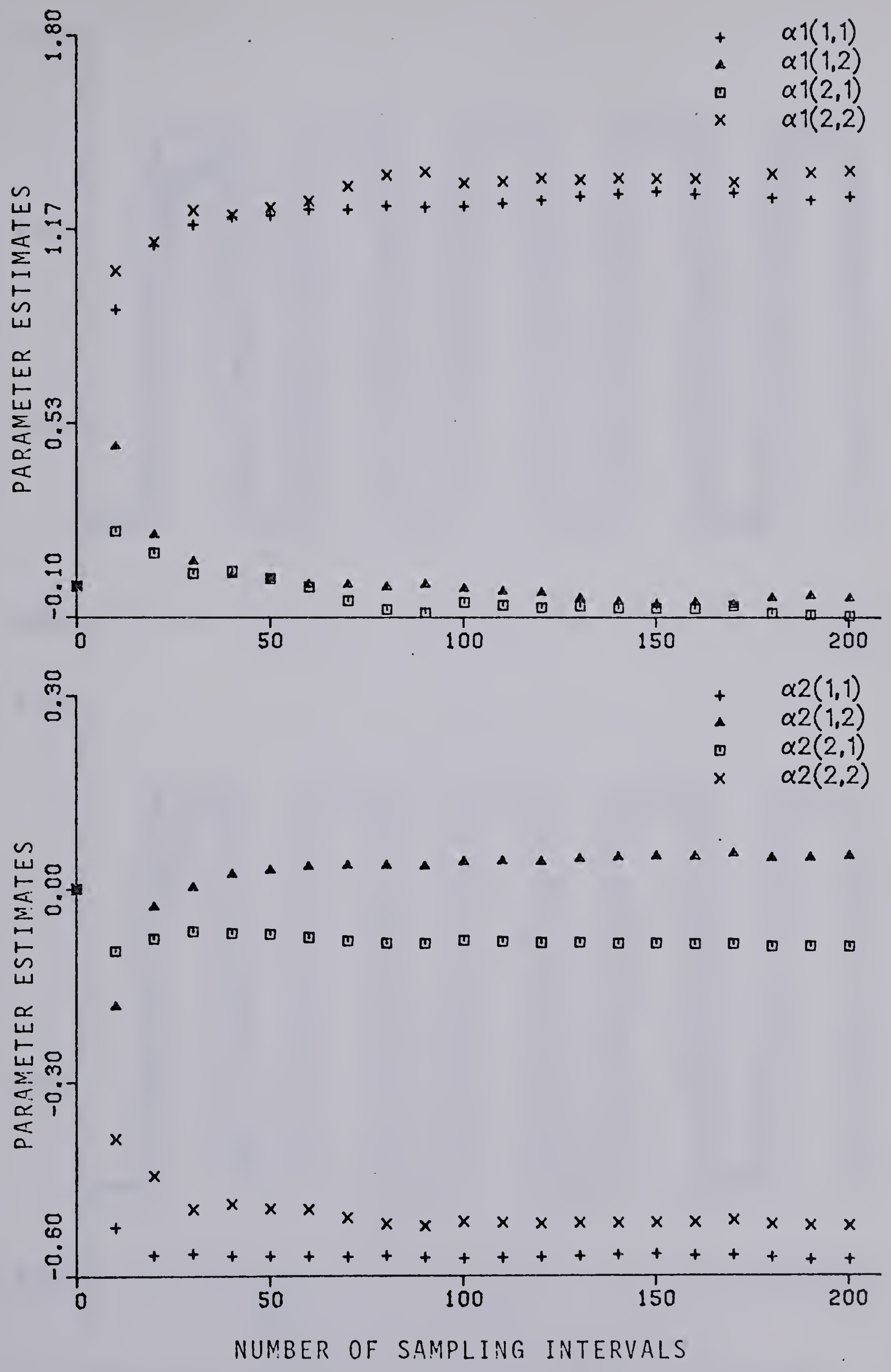


Figure 2d: Simulation of Keviczky second order system. Behaviour of the α parameter estimates.

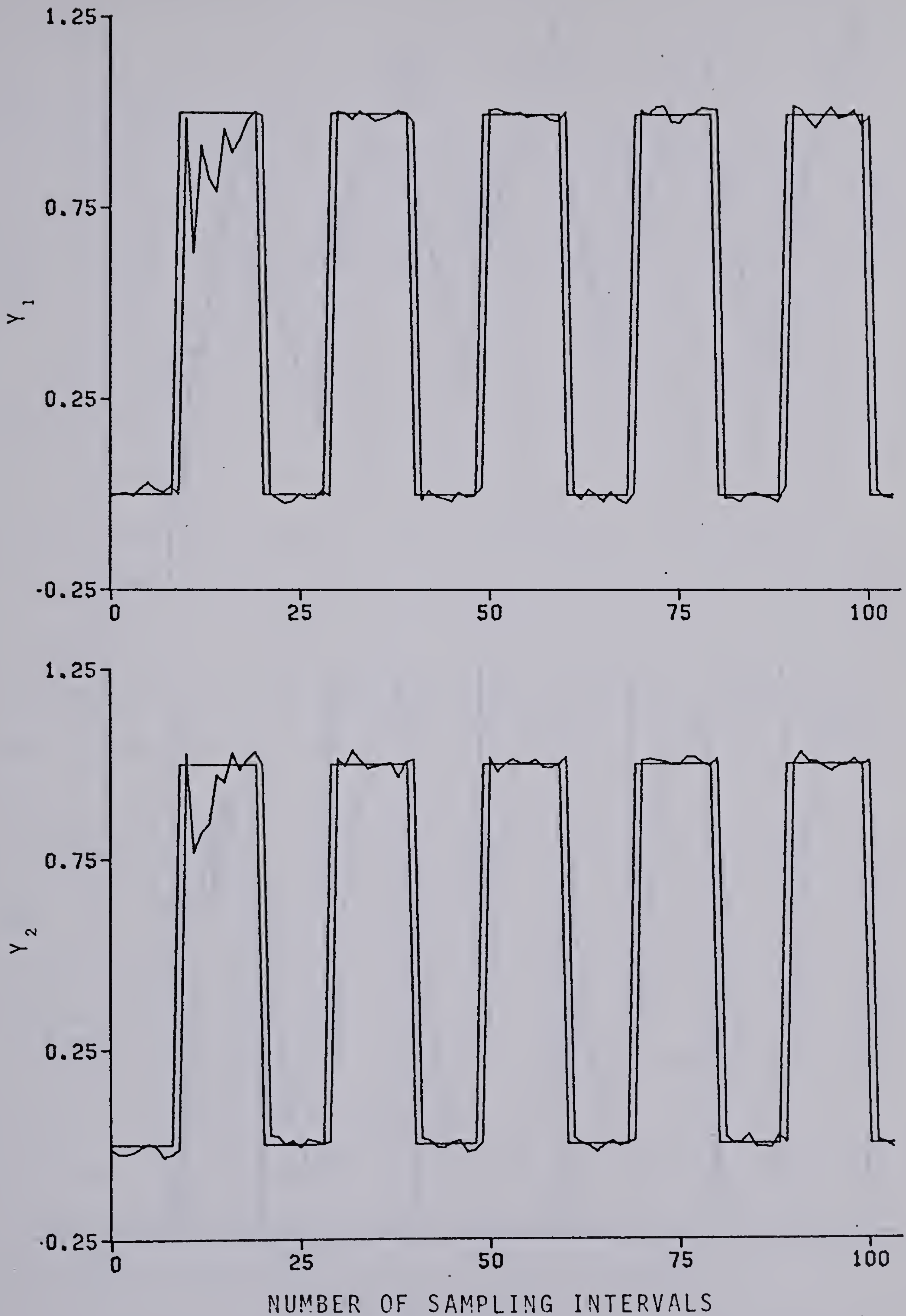
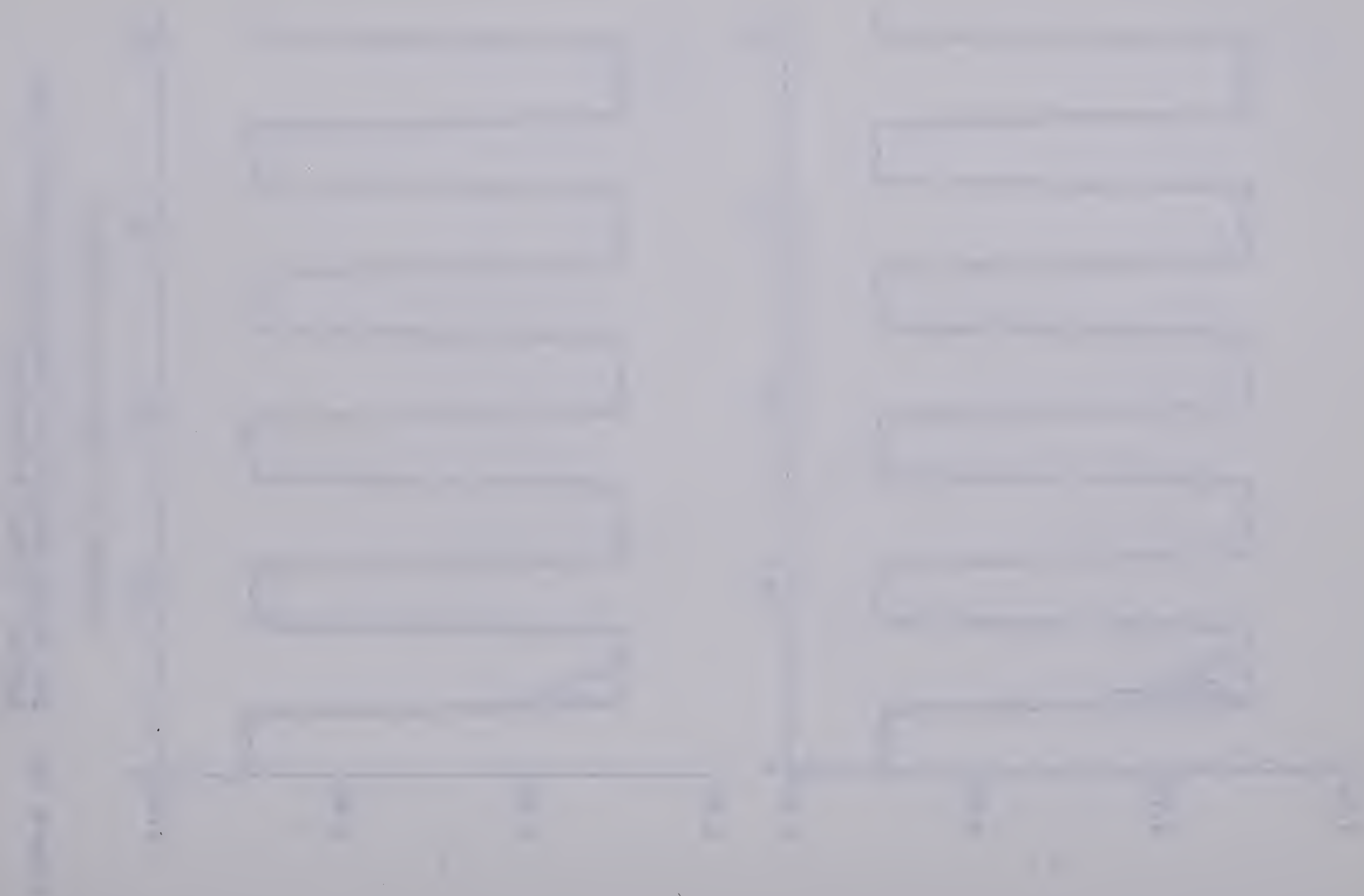


Figure 3a: Set point control of Keviczky second order system. Output behaviour.



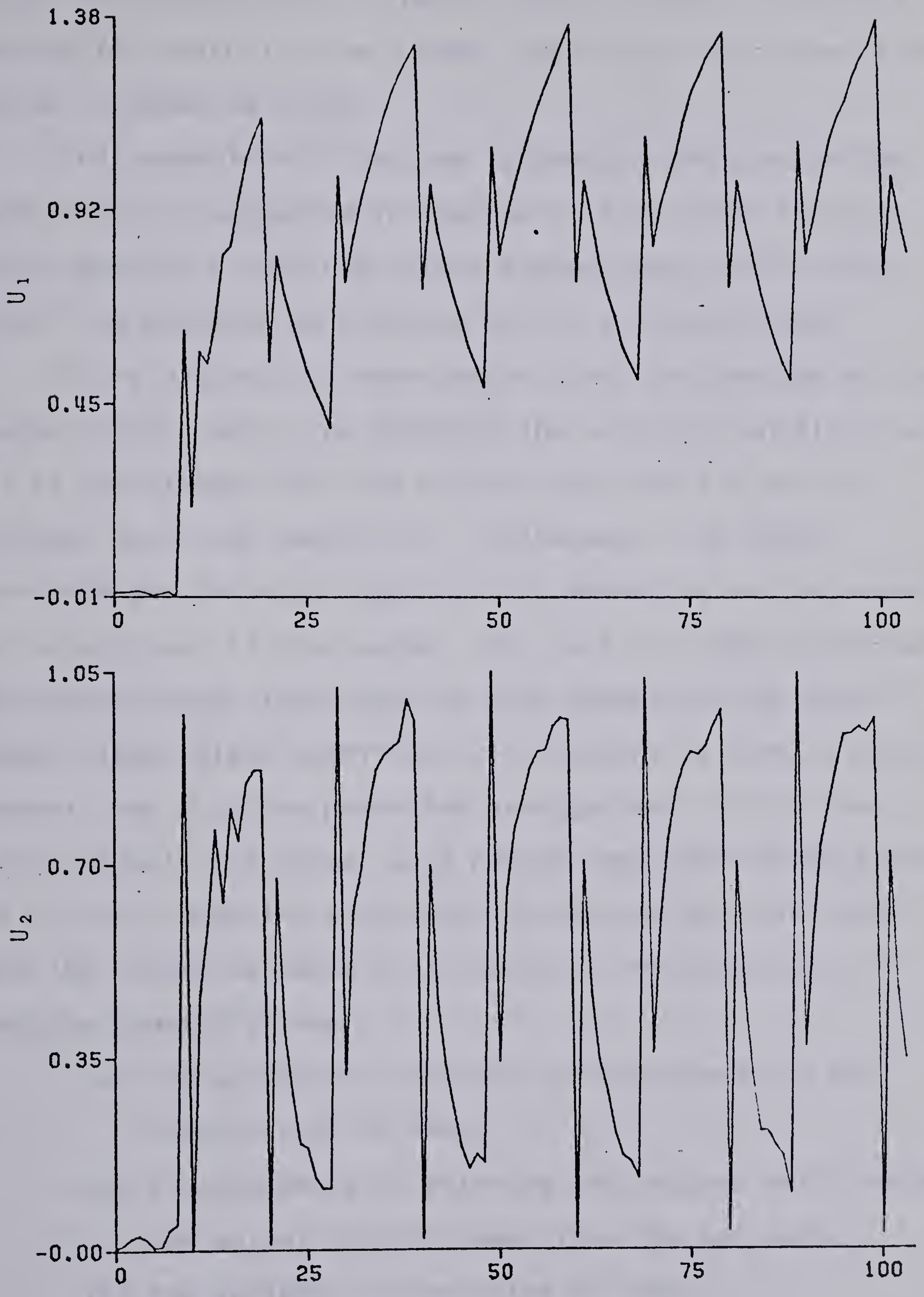


Figure 3b: Set point control of Keviczky second order system. Input behaviour.

input variables under set point control. After an initial period the control is very good; again the covariance of the noise is taken as $0.01\underline{\underline{I}}$.

This example will be used to demonstrate some of the modifications suggested in Chapter 3. First, the technique which ensures a required output average over finite time (RAFT) as proposed by Keviczky et al is investigated.

It is claimed by these workers that the average of the output might improve by changing the set point on-line, but it is not stated that this may only be true for certain process and noise conditions. Furthermore, the output variance can increase significantly depending on the amount of noise input to the system. This is to be expected because the noise causes the output to vary around the set point under steady state conditions. The average at every sampling moment, and thus the predicted average over finite time, will reflect this noise. As a result the modified set point will vary around the actual set point, and both the input and the output variance will increase. The technique will only be beneficial when:

(a) the parameters have not yet converged and the output error is large

(b) a disturbance is entering the process which causes the output to drift away from the set point.

(c) the variance of the noise is large.

(d) the setpoint is changed.

In these cases, more improvement is obtained by varying the

set point if the time over which the average is taken is increased.

To compare self-tuning controllers, which have the additional feature of ensuring a required output average over a finite time, with ordinary self-tuning controllers, the sum of the squares of the average output error after N samples is calculated; this will be called v_i for output i. The accumulated output variance is called AIE_i. Thus:

$$v_i = \sum_j [y_{a_i}(jN) - y_{r_i}]^2$$

$$AIE_i = \sum_t [y_i(t) - y_r]^2$$

Table I compares v and AIE for different noise covariances $\sigma \underline{\underline{I}}$. The set point is one and all initial parameters are zero. In all cases the parameters converged to the values corresponding to those of the minimum variance controller. As can be seen from Table I, the average error is indeed smaller using RAFT but it is not obvious whether the reduction takes place in all time periods or only during the initial period, when the parameter estimates are changing rapidly by a significant amount. Table II shows how the average error of the first output signal over N multiples of the sample interval behaves in time for the ordinary self-tuning regulator (STR) and for the self-tuning regulator which ensures a required average over finite time (STR-RAFT). It can be seen now that the technique will give

Table I: Variance and average error of the two output signals over 100 sample instants, for N=10.

RAFT	σ	AIE_1 (100)	AIE_2 (100)	V_1	V_2
NO	0.0	1.308	1.676	0.036561	0.005387
YES	0.0	1.531	2.191	0.000036	0.000010
NO	0.01	1.217	1.227	0.029219	0.008949
YES	0.01	1.370	1.437	0.000100	0.000004
NO	0.1	3.906	7.827	0.016443	0.156050
YES	0.1	5.809	11.354	0.001319	0.000560
NO	1.0	1403.800	460.246	7.762279	2.602635
YES	1.0	1268.845	496.822	1.118368	0.120644

Table II: Absolute value of the average error in percent after T sample intervals for N=10.

	σ=0.01		σ=1.0	
T	STR	STR-RAFT	STR	STR-RAFT
10	17.09	0.92	252.72	100.21
20	0.35	0.18	85.01	4.31
30	0.46	0.04	73.80	2.80
40	0.17	0.08	1.94	13.02
50	0.55	0.20	23.36	15.63
60	0.17	0.17	0.40	19.81
70	0.08	0.01	11.92	0.29
80	0.18	0.12	2.00	11.06
90	0.22	0.20	19.65	14.97
100	0.36	0.28	51.68	30.55

a significant improvement in all time periods if the noise signal is large. But if the noise is small, it is only in the first period that the error reduces significantly. Table III shows that if N increases, the error will reduce even more when using RAFT.

When a disturbance enters the system, changing the set point will also improve the average error. Table IV gives the average error when a sawtooth disturbance with a period of 30 sampling intervals and magnitude 1.5, enters the system.

If the set point of y_1 is changed from one to two, the average error over 10 sampling intervals is equal to 4.99% of the set point in the first time period and approximately 0.015% in subsequent periods when using STR. Using RAFT technique, this becomes 0.026% and 0.0015% respectively, which shows that the improvement occurs mostly in the first period after the change in set point.

In the study of single input - single output self-tuning regulators, Åström [3] gives a rule for fixing β_0 and shows that the output variance is minimal if β_0 is set equal to the actual value. In the multivariable case it is necessary to select $\underline{\underline{\beta 0}}$ in such a way that the eigenvalues of the matrix $[\underline{\underline{I}} - \underline{\underline{B0}}\,\underline{\underline{\beta 0}}^{-1}]$ are smaller than one. Table V gives the accumulated output variance for different choices of $\underline{\underline{\beta 0}}$, and also for the case where $\underline{\underline{\beta 0}}$ is estimated. This shows that unless a very good estimate of $\underline{\underline{B0}}$ is available, which is not often the case for an actual process, it is

Table III: Absolute value of the average output error in percent after T sample intervals for N=100.

T	STR ($\sigma=1.0$)	STR-RAFT
100	13.14	3.12
200	13.02	0.51
300	0.50	0.86
400	14.93	0.47
500	4.98	0.88
600	5.43	0.73
700	10.06	2.27
800	10.38	1.10
900	10.86	0.88
1000	18.33	1.16

Table IV: Average output error in percent for a disturbance, N=30.

	y_1		y_2	
T	STR	STR-RAFT	STR	STR-RAFT
30	5.58	0.01	3.04	0.01
60	9.57	0.41	33.18	0.47
90	7.83	0.04	8.20	0.10
120	5.73	0.12	5.30	0.11
150	5.30	0.22	4.75	0.23
180	4.40	0.22	4.52	0.34

Table V: Variance of the output signals for different values of $\underline{\underline{\beta}}0$ when $\sigma = 0.01$

$\underline{\underline{\beta}}0$	μ	$AIE_1(100)$	$AIE_2(100)$
$\begin{vmatrix} 2.0 & -0.3 \\ 0.1 & 1.0 \end{vmatrix}$	0.0 0.0	1.217	1.227
$\begin{vmatrix} 2.0 & 0.0 \\ 0.0 & 1.0 \end{vmatrix}$	0.1225 -0.1225	80.483	16.417
$\begin{vmatrix} 0.5 & 0.0 \\ 0.0 & 0.5 \end{vmatrix}$	2.94 1.06	NO CONVERGENCE	
$\underline{\underline{\beta}}0$ ESTIMATED		1.935	1.526

much better to include the estimation of $\underline{\underline{B}}0$ in the algorithm. Also, if the system is nonlinear and $\underline{\underline{B}}0$ changes to a new value $\underline{\underline{B}}0'$ because of set point changes or a disturbance, it is better to let $\underline{\underline{\beta}}0$ converge to a new value, than to keep it fixed at its old value, because the eigenvalues of $[\underline{\underline{I}} - \underline{\underline{B}}0'\underline{\underline{\beta}}0^{-1}]$ might be larger than one.

4.3 System with time delay

The second system used for evaluation of the regulator is that given by Borisson [7]. The system, which is open loop unstable, can be expressed as:

$$\underline{y}(t) = \underline{\underline{A}}\underline{y}(t-1) + \underline{u}(t-2) + \underline{e}(t) + \underline{\underline{C}}\underline{e}(t-1) \tag{60}$$

$$\underline{\underline{A}} = \begin{vmatrix} 0.9 & -0.5 \\ -0.5 & 0.2 \end{vmatrix} \qquad \underline{\underline{C}} = \begin{vmatrix} -0.2 & -0.4 \\ 0.2 & -0.8 \end{vmatrix}$$

$$E\{\underline{e}(t)\underline{e}^T(t)\} = \begin{vmatrix} 0.1 & 0.1 \\ 0.1 & 0.2 \end{vmatrix}$$

The optimal control law is given by:

$$[\underline{\underline{I}} + \tilde{\underline{\underline{F}}}q^{-1}]\underline{u}(t) = -\tilde{\underline{\underline{G}}}\underline{y}(t) \tag{61}$$

$$\tilde{\underline{\underline{F}}} = \begin{vmatrix} 1.4143 & 0.98571 \\ -1.1857 & -1.3143 \end{vmatrix} \qquad \tilde{\underline{\underline{G}}} = \begin{vmatrix} 0.78 & -0.51 \\ -0.41 & 0.33 \end{vmatrix}$$

The self-tuning algorithm estimates the parameters of the prediction model:

$$\underline{y}(t) = \underline{\underline{A}}\underline{y}(t-2) + \underline{\underline{B}}0\underline{u}(t-2) + \underline{\underline{B}}1\underline{u}(t-3) + \underline{\varepsilon}(t) \tag{62}$$

The optimal values are: $\underline{\underline{A}} = \tilde{\underline{\underline{G}}}$, $\underline{\underline{B}}0 = \underline{\underline{I}}$, $\underline{\underline{B}}1 = \tilde{\underline{\underline{F}}}$.
Borisson showed that, for a fixed value of $\underline{\underline{\beta}}0$ ($=\underline{\underline{I}}$), the $\underline{\underline{\beta}}1$ parameters have not converged to the optimal values even after 10^5 sample instants, but the control performance was very good after 20 sampling intervals. The behaviour of the system for the first 830 intervals can be seen from Figures 4a, 4b and 4c. Borisson showed the same behaviour for the parameter estimates but over a period of 10^5 sampling intervals. Due to the noise, the variances of input and

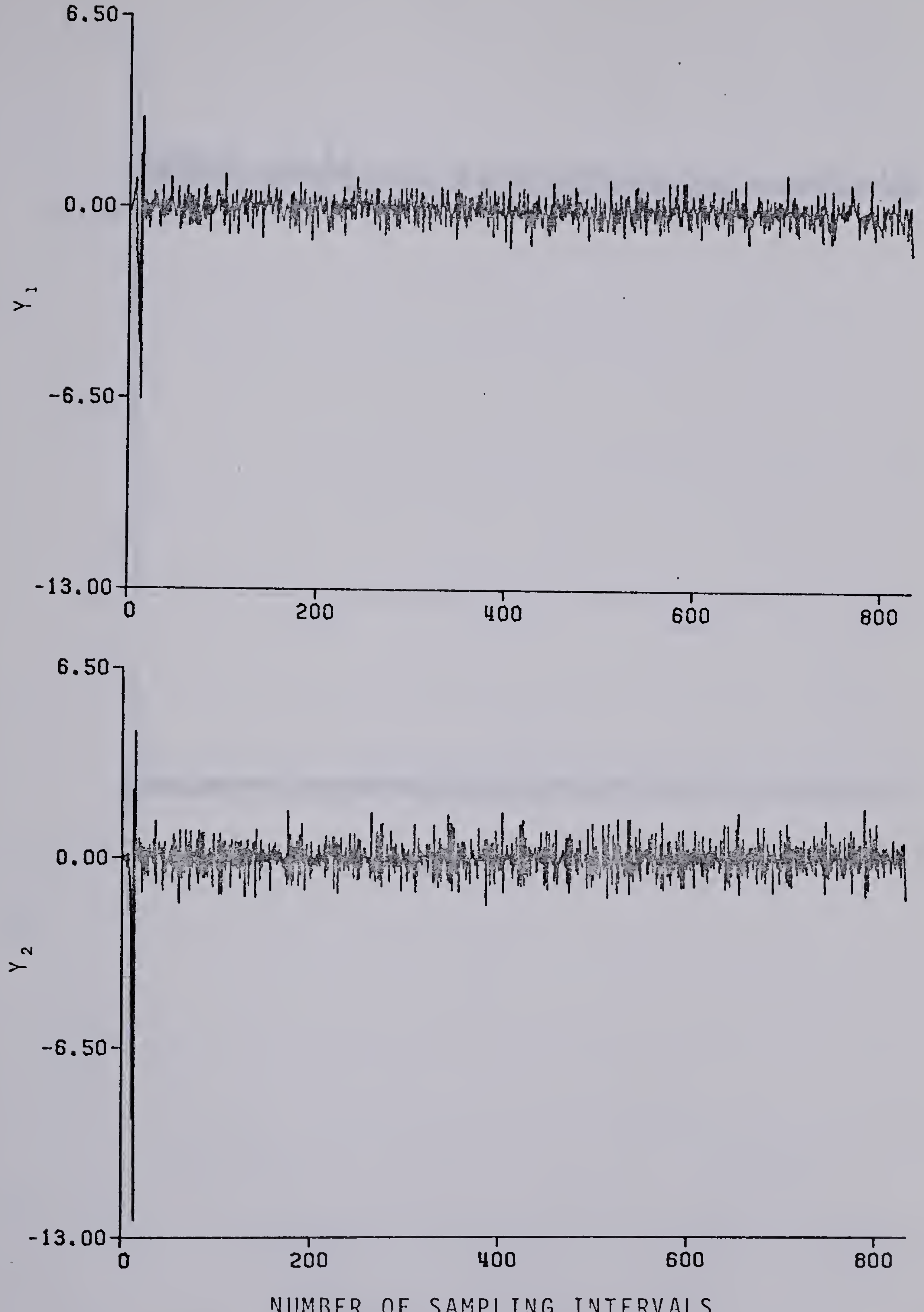
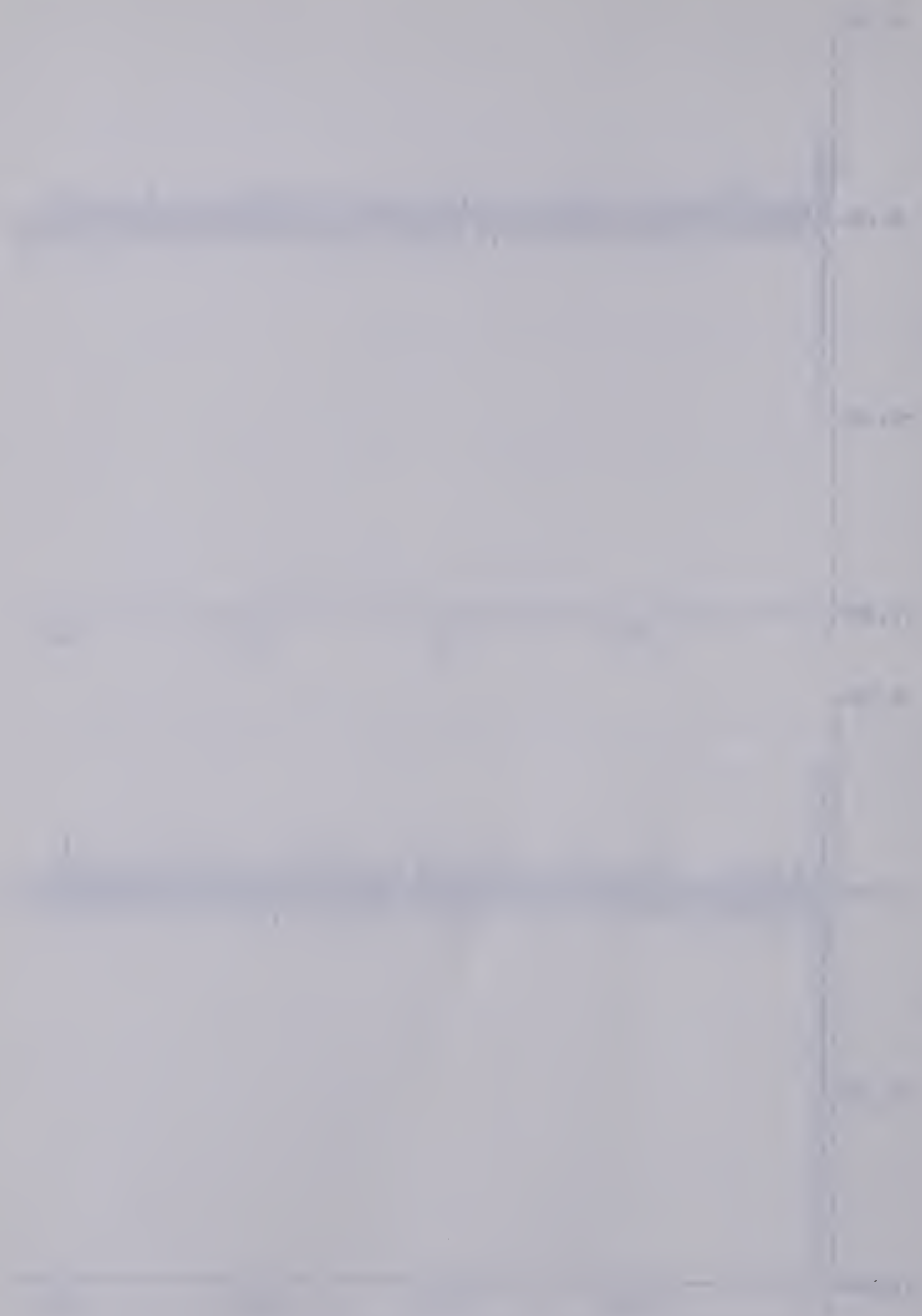


Figure 4a: Simulation of Borisson system. Output behaviour.



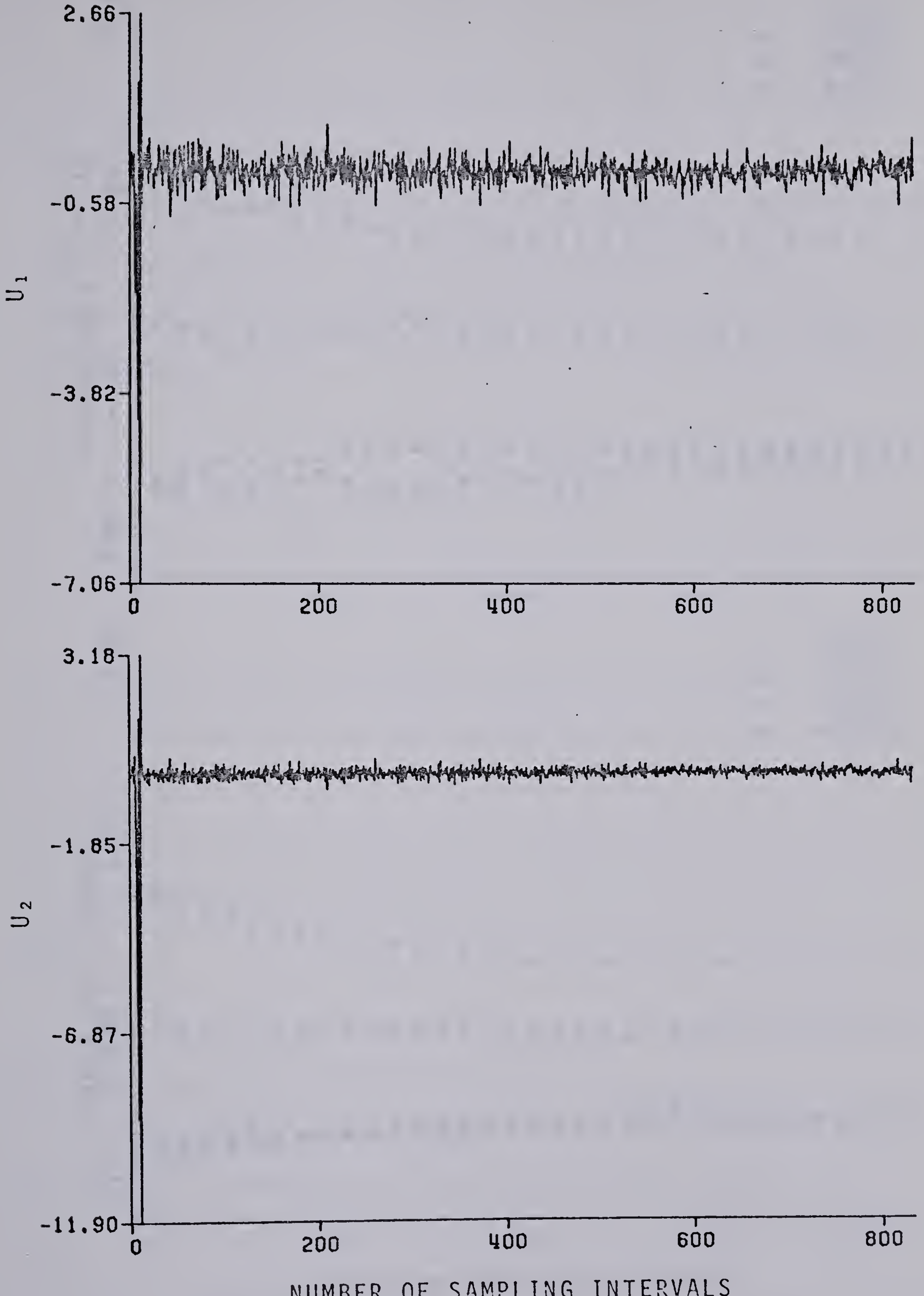
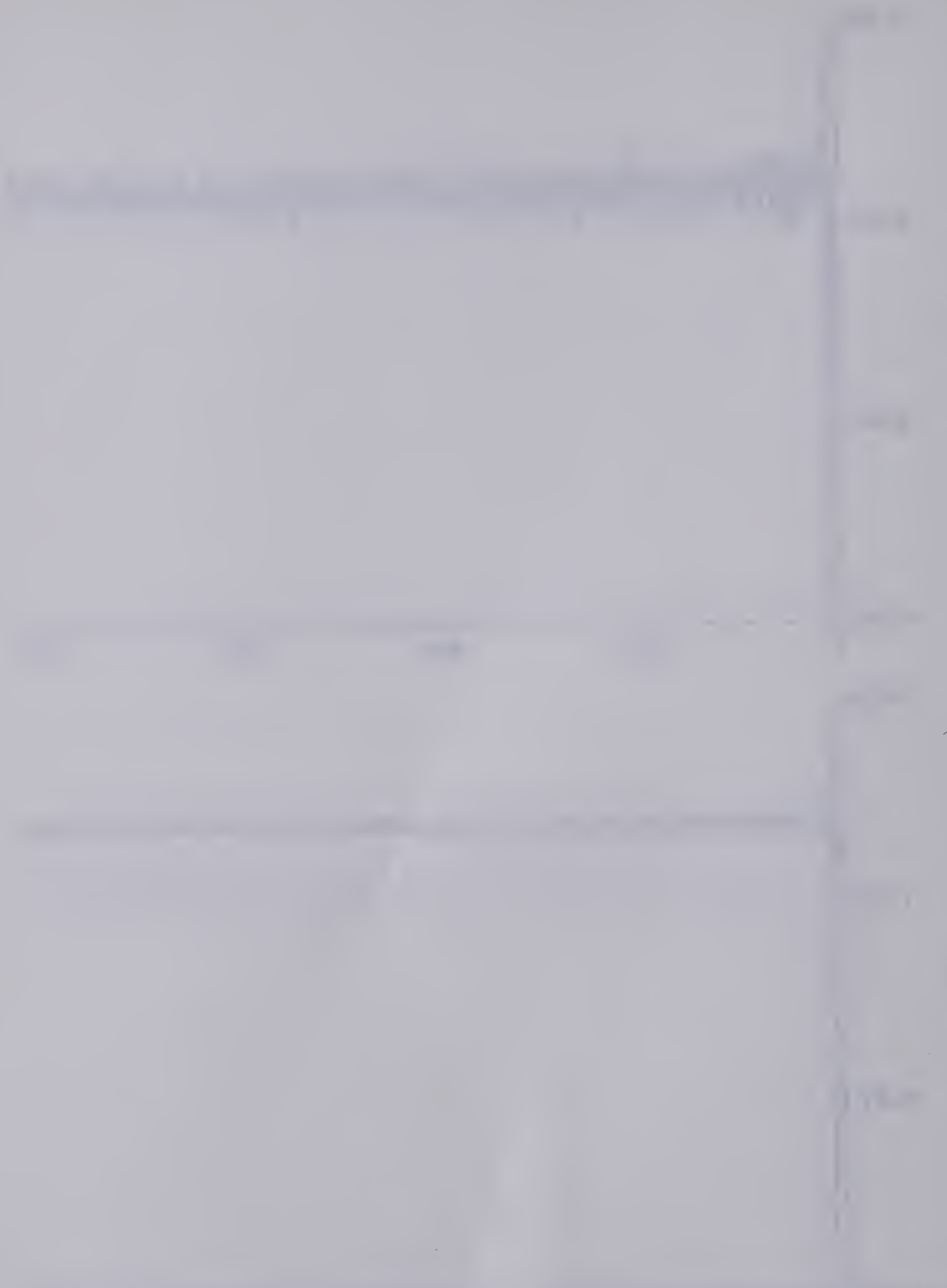


Figure 4b: Simulation of Borisson system. Input behaviour.



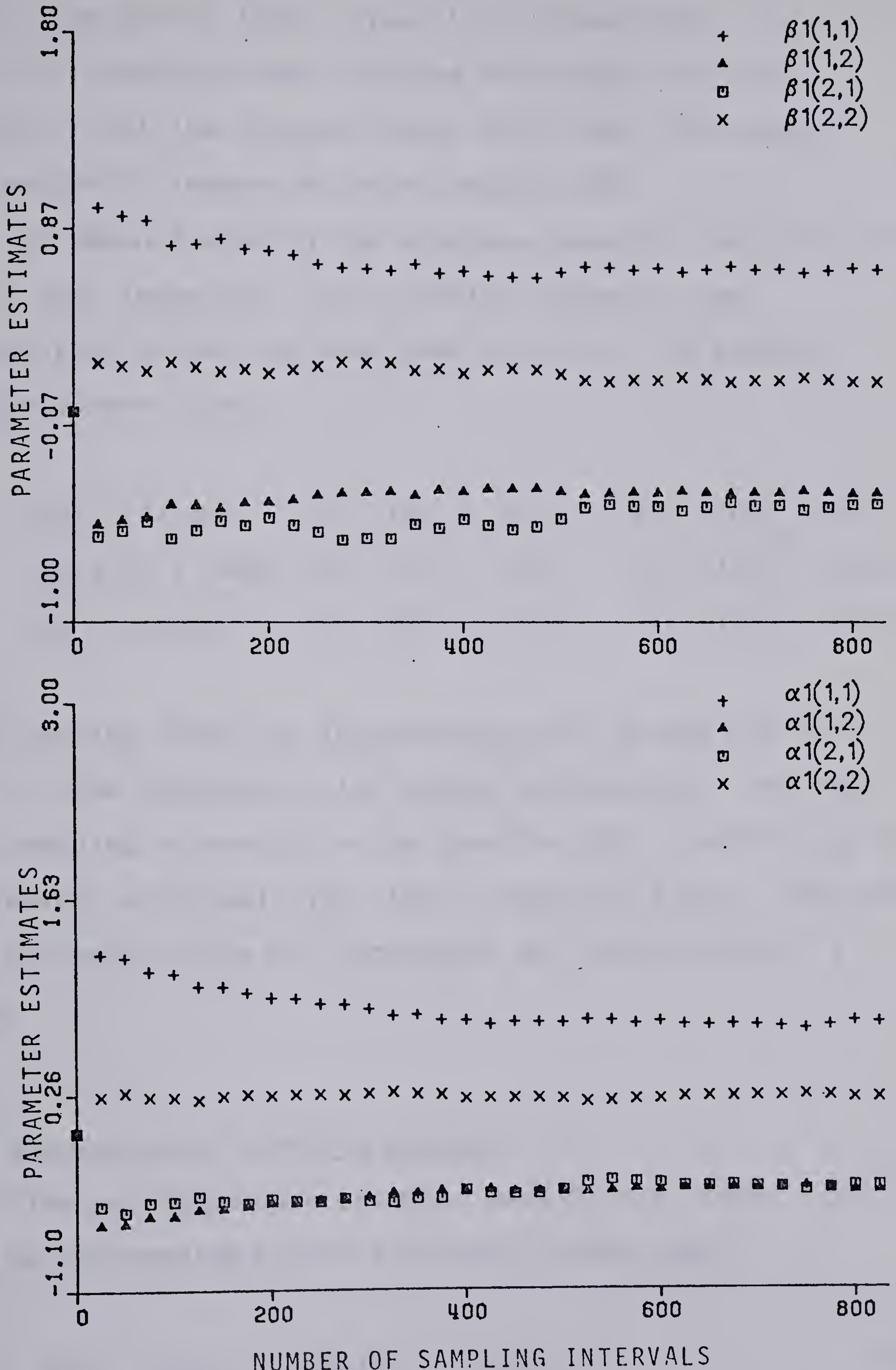


Figure 4c: Simulation of Borisson system. Parameter estimates.

output are fairly large. Simulations showed that the use of the RAFT technique does increase the output variance slightly, but the average output error over 100 sample instants will improve by approximately 25%.

As demonstrated in the previous example, the choice of $\underline{\underline{\beta}}0$ is very important. The following values of the accumulated output variance over the first 100 sample instants were found:

$\underline{\underline{\beta}}0=\underline{\underline{I}}$ (fixed)	$AIE_1(100) = 107.1$	$AIE_2(100) = 307.2$
$\underline{\underline{\beta}}0=0.5\underline{\underline{I}}$ (fixed)	$AIE_1(100) = 1381$	$AIE_2(100) = 3278$
$\underline{\underline{\beta}}0$ estimated	$AIE_1(100) = 44.81$	$AIE_2(100) = 42.19$

The starting value for the estimation of $\underline{\underline{\beta}}0$ was the unit matrix. The increase in the output variance over the next 100 sampling intervals is the same for $\underline{\underline{\beta}}0 = \underline{\underline{I}}$ and for $\underline{\underline{\beta}}0$ is estimated, and equals 18.7 for y_1 and 32.5 for y_2. Even when the parameters have not converged, set point control is good.

<u>4.4 Simulation of a mixing process</u>

The mixing process example, used by N.M. Koivo [12], can be represented by the following linear model:

$$\underline{y}(t) + \underline{\underline{A}}\underline{y}(t-1) = \underline{\underline{B}}\underline{u}(t-1) + \underline{\underline{C}}\underline{e}(t) \tag{63}$$

$$\underline{\underline{A}} = \begin{vmatrix} 0.98 & 0.0 \\ -0.01 & 0.98 \end{vmatrix} \qquad \underline{\underline{B}} = \begin{vmatrix} 0.012 & 0.012 \\ -0.023 & 0.023 \end{vmatrix}$$

$$\underline{\underline{C}} = \begin{vmatrix} 0.02 & 0.7 \\ -0.9 & -0.2 \end{vmatrix}$$

As stated by Koivo, since $\underline{\underline{B}}$ has very small elements, this will give rise to large variations in the control signal. The simulation results presented by Koivo exhibit bang-bang control between the limits +10.0 and -10.0 for different set points of y_1 and y_2. The dependent variables, although denoted as the control signals, are presumably deviations as can be seen from examining the gain for this system:

$$[\underline{\underline{I}} + \underline{\underline{A}}]^{-1}\underline{\underline{B}} = \begin{vmatrix} 0.00606 & 0.00606 \\ -0.01161 & 0.01162 \end{vmatrix}$$

The large gain means that, if both set points are 1.0, u_1 must have an average of 39.5 and u_2 must be 125.5. Simulations of this system were done for the following set point changes:

$T<100$: $y_{1s}=0.0$, $T<300$: $y_{1s}=2.0$, $T>300$: $y_{1s}=3.0$

$T<200$: $y_{2s}=0.0$, $T<300$: $y_{2s}=5.0$, $T>300$: $y_{2s}=4.0$

where T denotes the number of sampling intervals. The control signal was limited between +450 and -450. The covariance of the noise was $0.3\underline{\underline{I}}$ and $\underline{\underline{\beta}}$ was estimated. Figures 5a, 5b and 5c show a simulation run with $\underline{\underline{\beta}}$ initially equal to the unit matrix and a forgetting factor of 0.99.

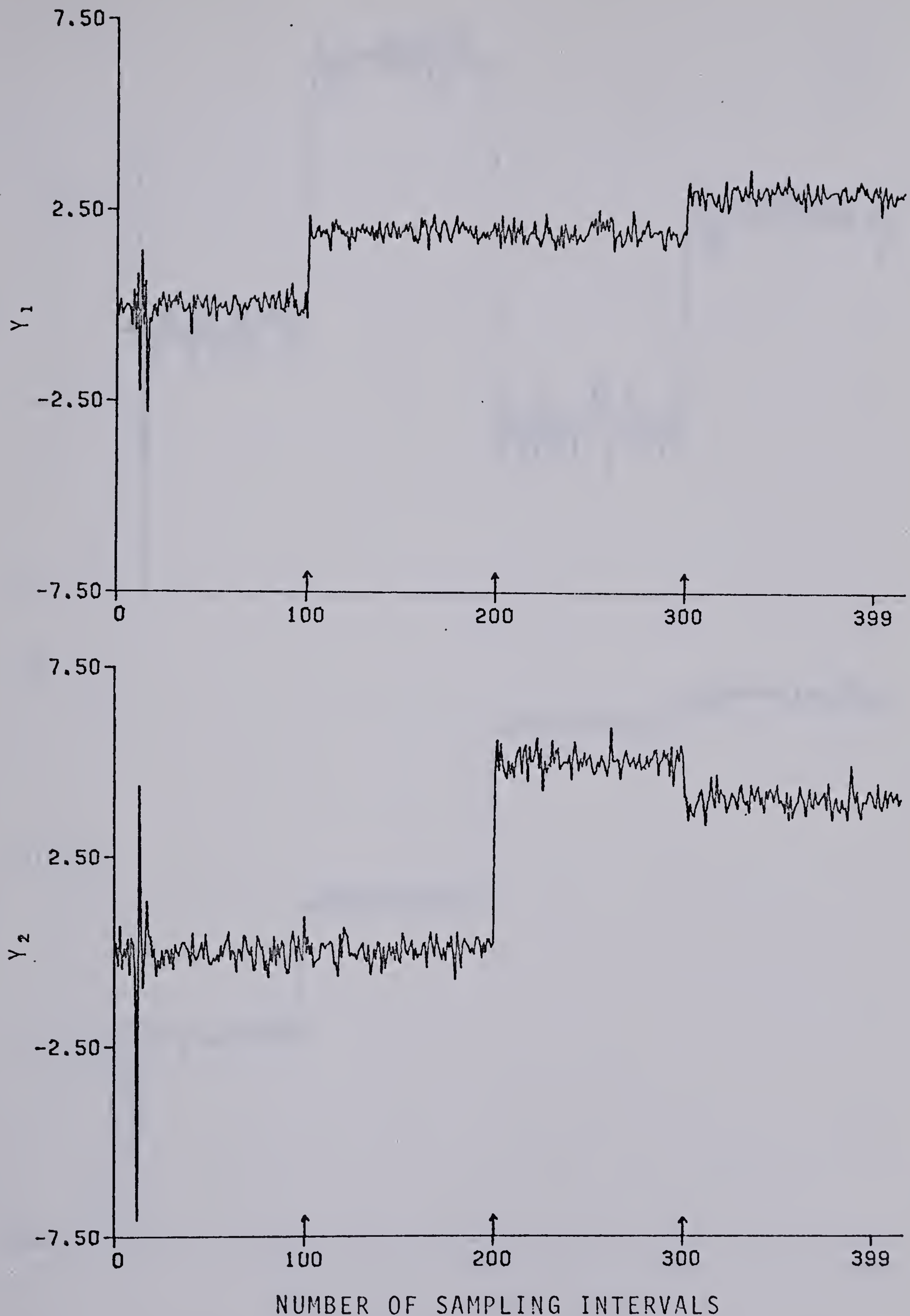
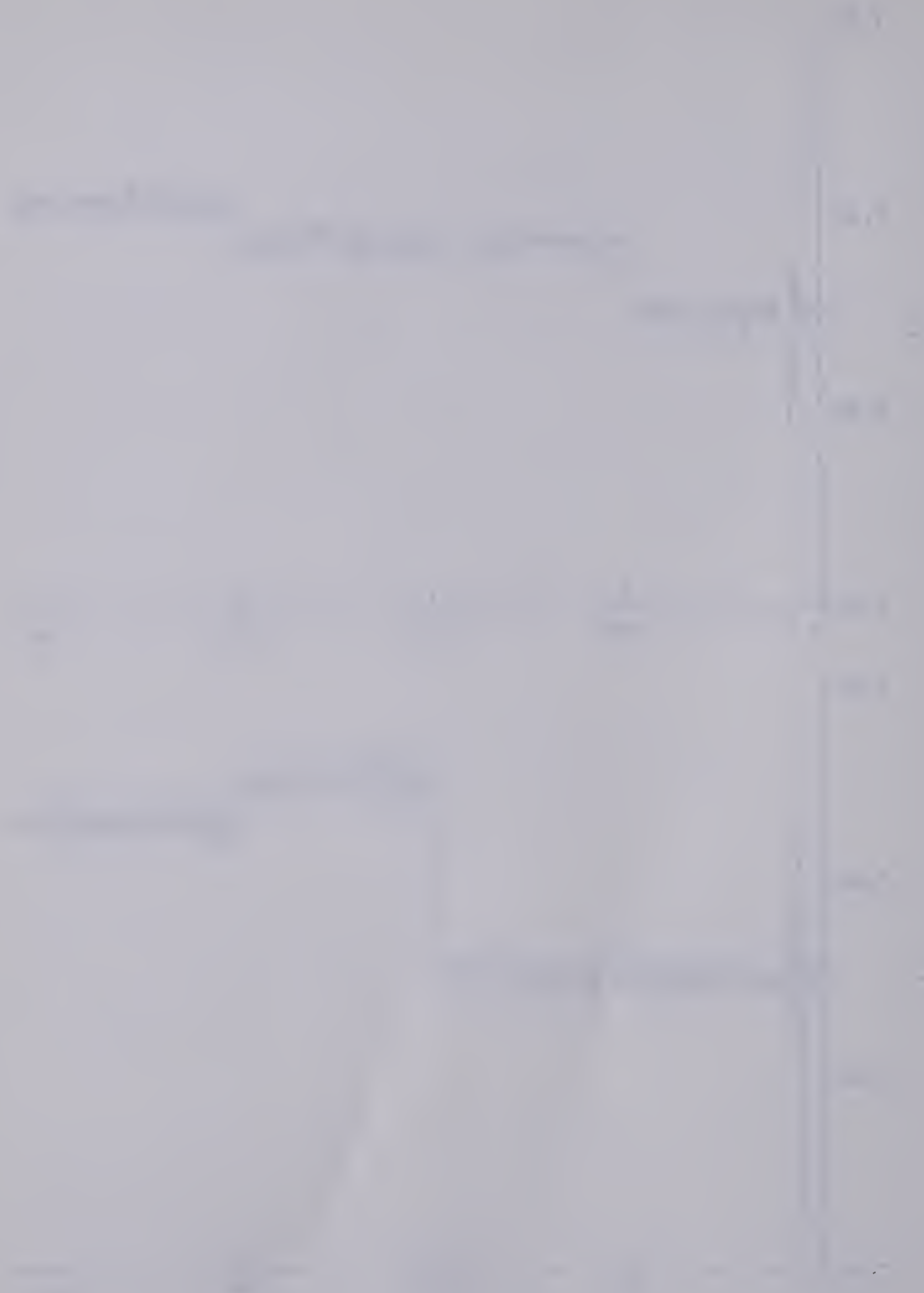


Figure 5a: Simulation of a mixing process. Output behaviour.



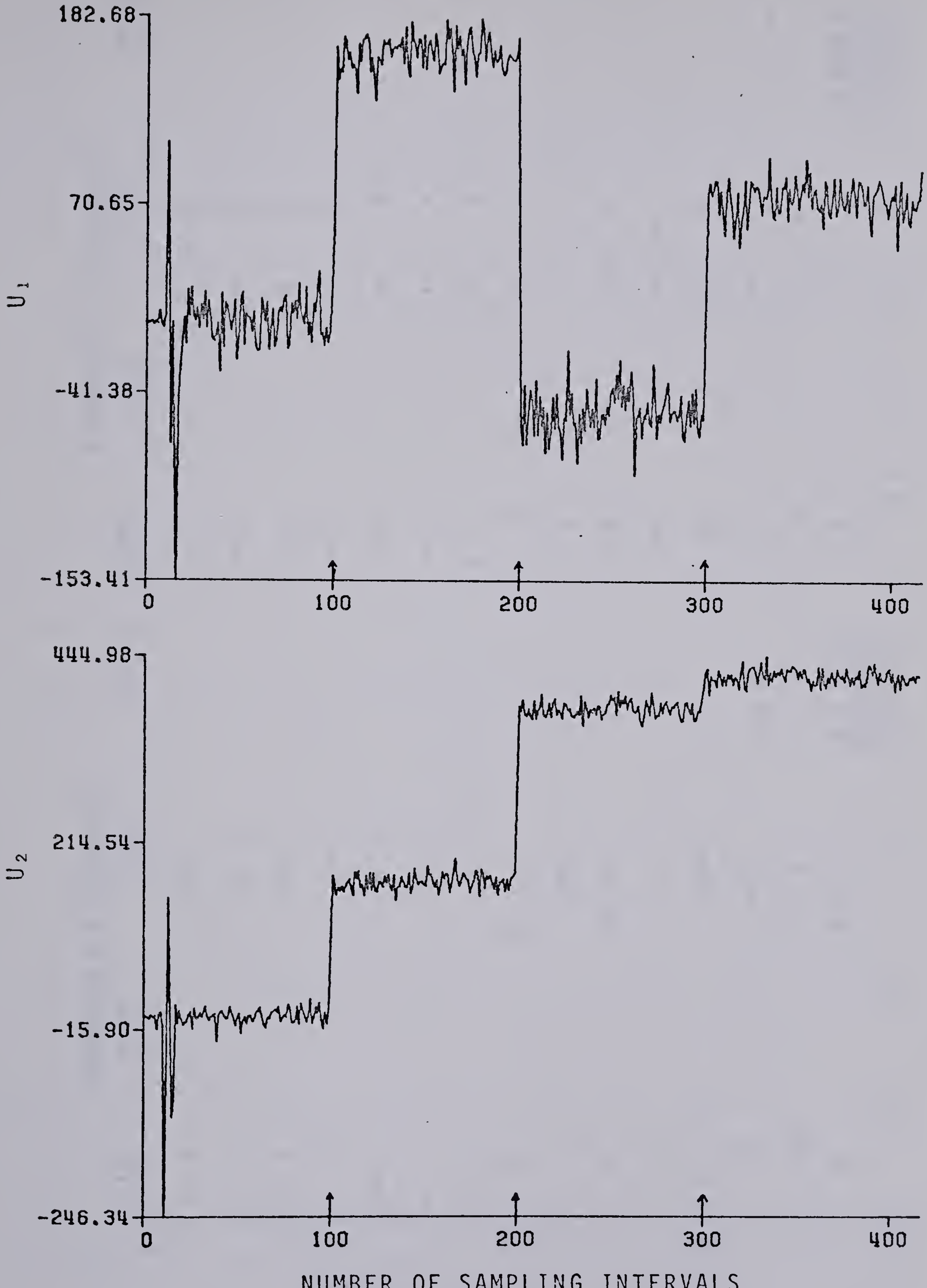


Figure 5b: Simulation of a mixing process. Input behaviour.

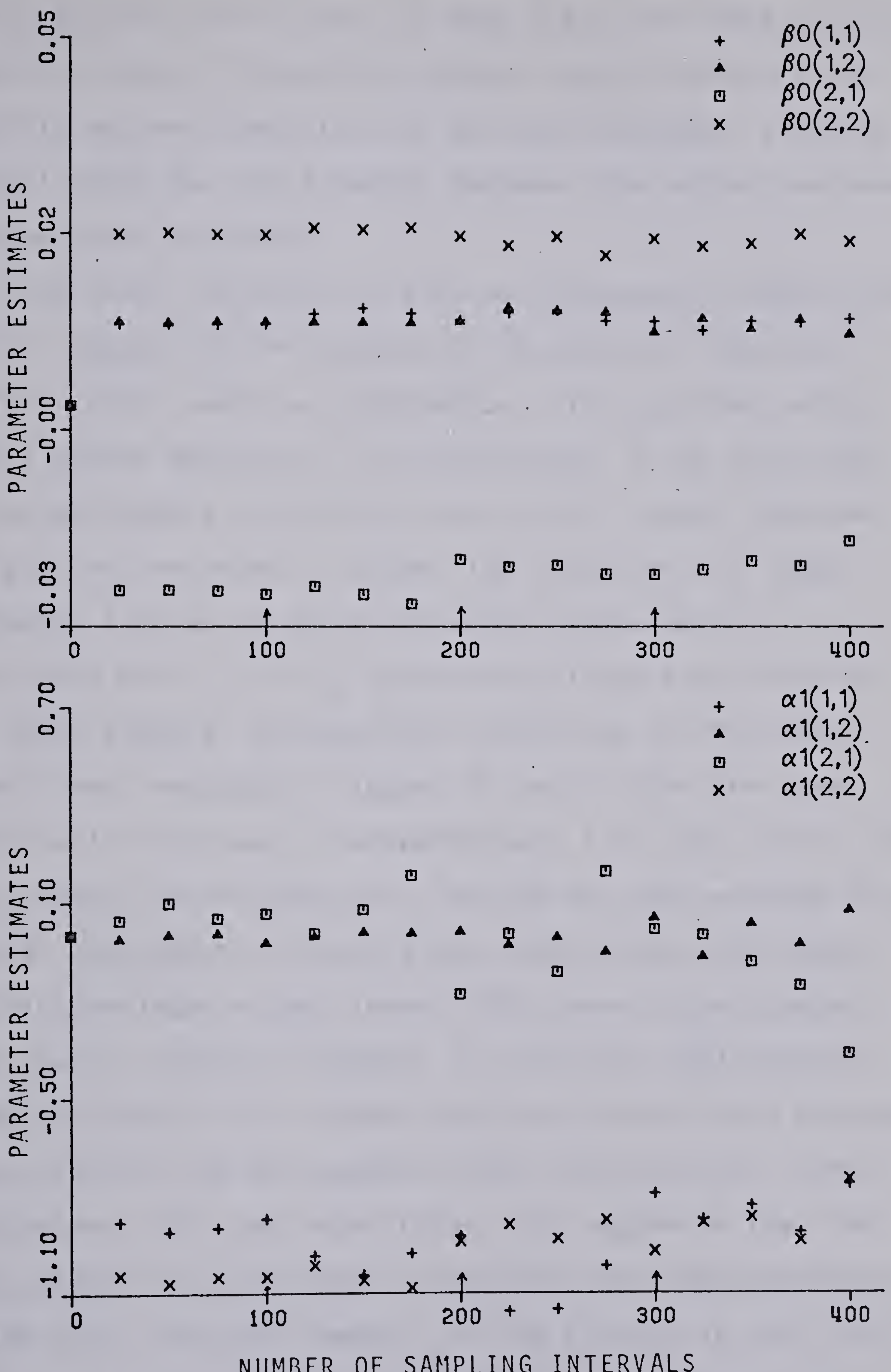


Figure 5c: Simulation of a mixing process. Parameter estimates.

The forgetting factor does not have a big influence on the output variance. A forgetting factor lower than one gives slightly quicker adaptation to set point changes. A better initial value for the $\underline{\underline{\beta}}$ matrix improves the output variance and the input variance.

The input variance can also be improved by limiting the control signal, or the changes in the control signal. A similar effect would be obtained by filtering the control signal before applying it to the process, or by filtering the output before calculating the control values. Because the gain of the noise is large, the inputs have a large variance. Figures 6a and 6b show that noise with a covariance matrix of $0.3\underline{\underline{I}}$ causes very large excursions in the input signals, although the variations in the output signals are reasonable. Figures 7a and 7b show the same system with the inputs limited between +10.0 and -10.0. This will reduce the variance of u_1 by 60% and the variance of u_2 by 35%. The output variance stays approximately the same. The disadvantage is that these limits have to be changed when the set point is changed. In practical applications large variances of the input signal are undesirable because most actuators can not support rapid fluctuations. Clarke and Gawthrop [8], and later Koivo [12] suggested that the input signals be penalized by including the input variances in the cost function. However, if the penalty is too large, the input signals might not be able to adapt themselves to set point changes or disturbances, and offset might occur as

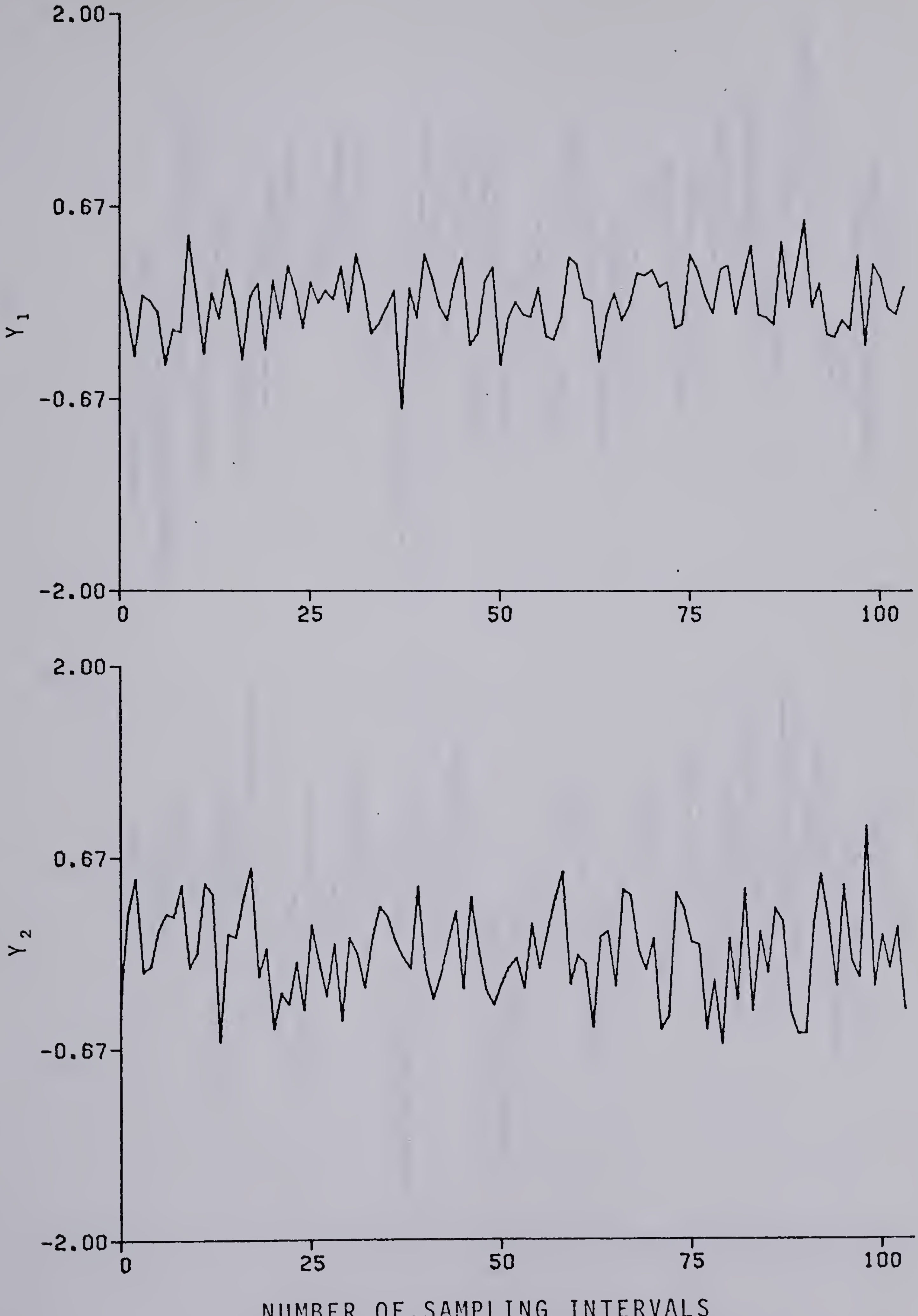


Figure 6a: Simulation of a mixing process. Output behaviour when the inputs are not limited.

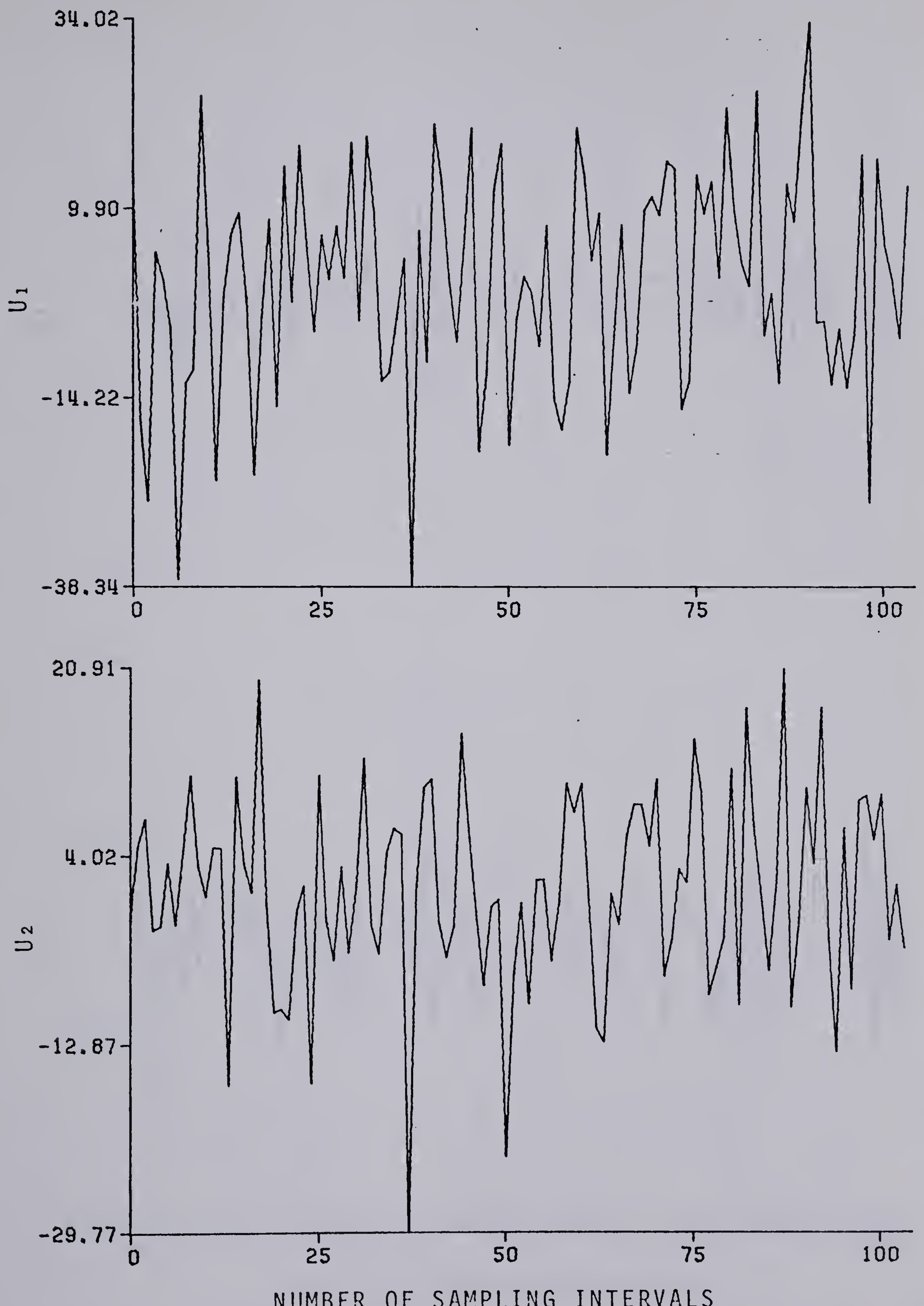


Figure 6b: Simulation of a mixing process. Input behaviour when the inputs are not limited.

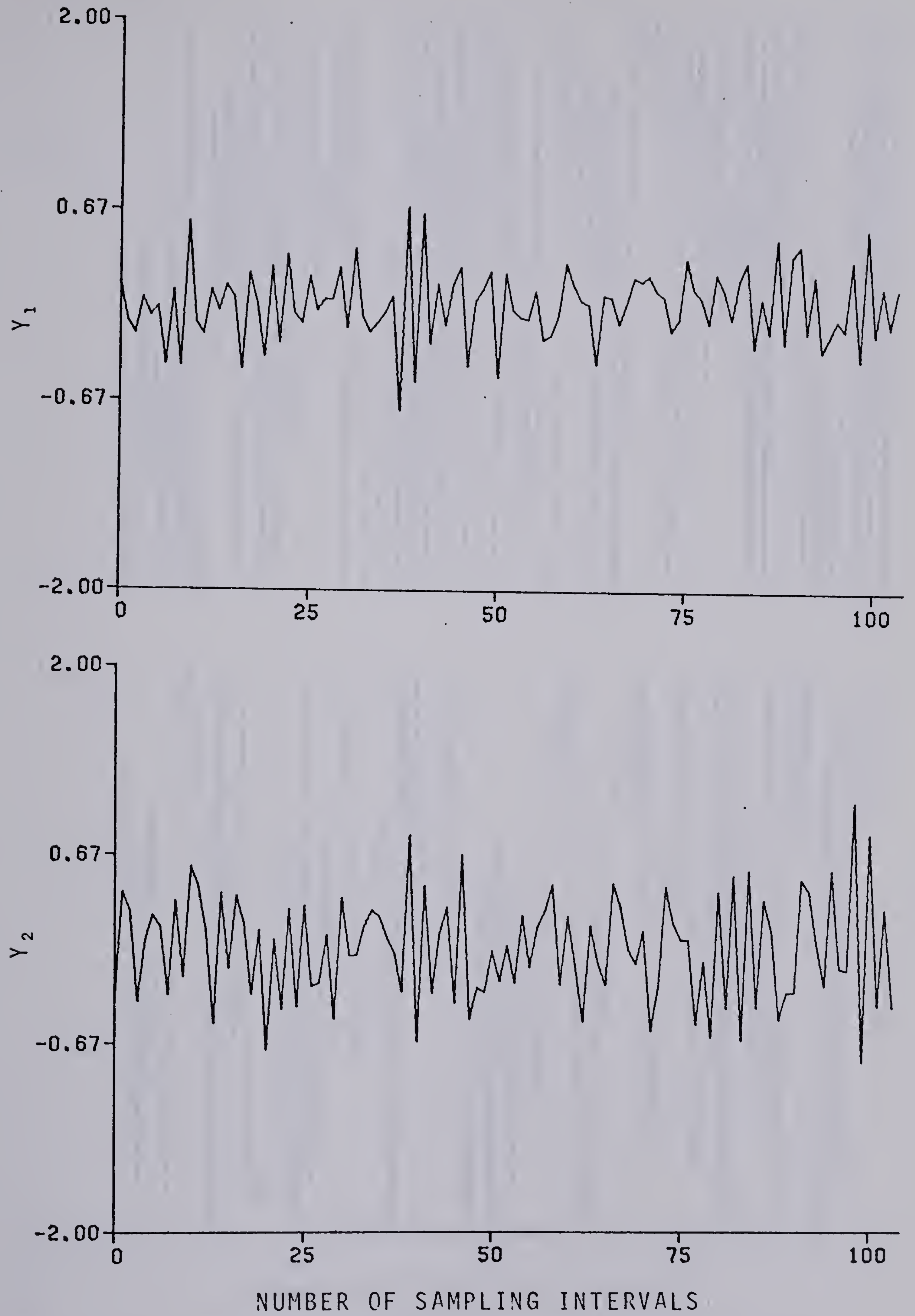


Figure 7a: Simulation of a mixing process. Output behaviour for limited input signals.

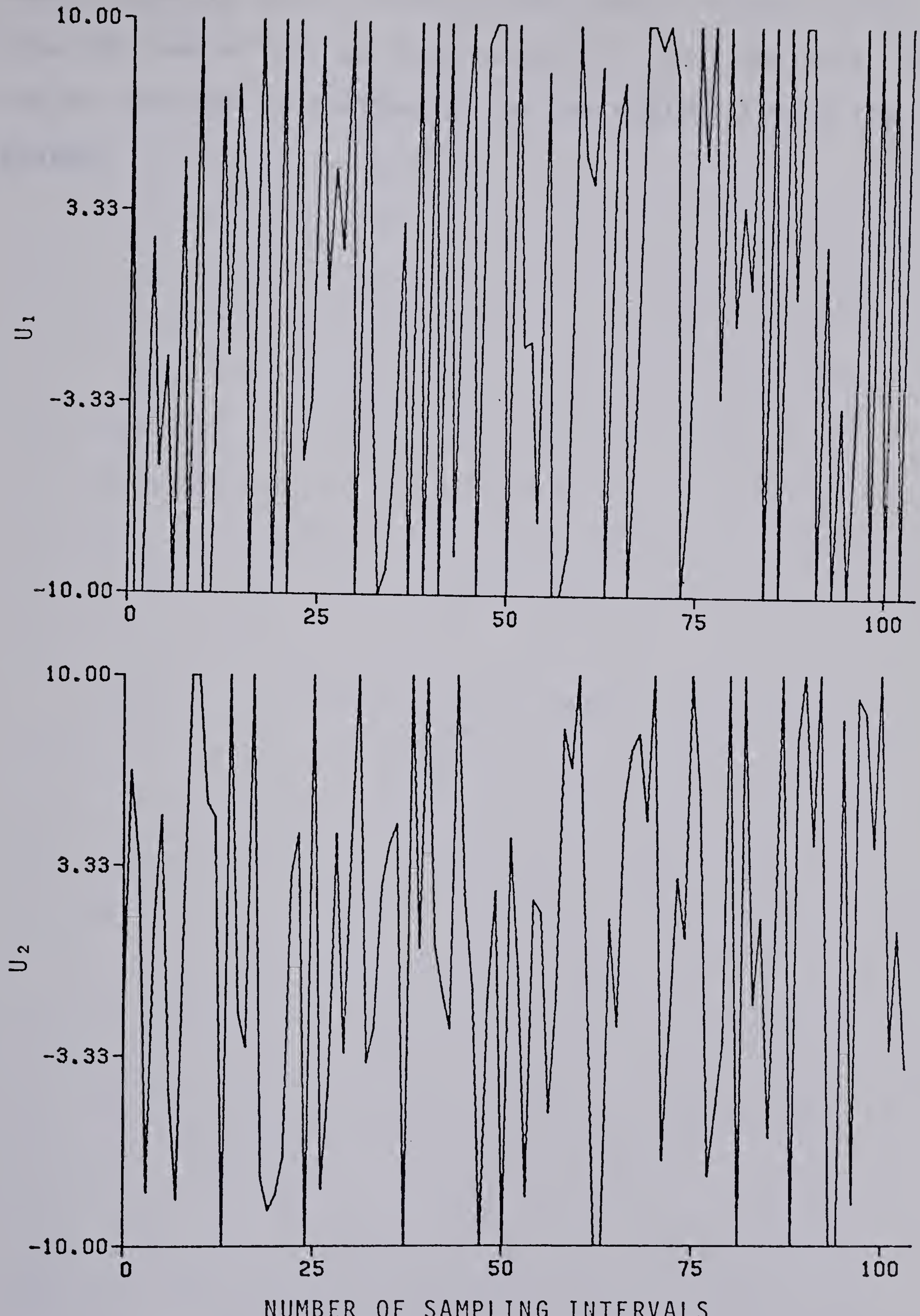


Figure 7b: Simulation of a mixing process. Input behaviour for limited input signals.

demonstrated by Koivo. Limiting the control signals could have the same effect so the decrease in input variance that can be obtained is limited by the controllability of the system.

5. SIMULATION STUDY OF A TRANSFER FUNCTION MODEL OF A BINARY DISTILLATION COLUMN

5.1 Description of the model.

The dynamic behaviour of a binary distillation column can be represented in terms of a transfer function model as shown in Figure 8. The top and bottom product compositions are y_1 and y_2 respectively, u_1 is the reflux flow and u_2 is the steam flow. The feed flow F will be treated as a measurable disturbance to the system and its effect on the outputs is modelled by the transfer functions L_1 and L_2. The transfer functions relating the inputs and outputs are given by G_{11} , G_{12} , G_{21} and G_{22}.

Table VI [18] gives a set of typical asymmetrical transfer functions which will be used to simulate column behaviour. By introducing a zero-order hold circuit and taking the Z-transform, a discrete or sampled-data representation is obtained. An important parameter that must be selected is the sampling time. A small sampling time is rejected because this might result in a nonminimum phase system and also because the time delay would be much bigger than the sampling time. Control in this case would be very difficult or impossible. Simulations showed that sampling times lower than 4 minutes did not give stable control. Sampling times of 5 and 7 minutes were employed in the simulations presented in this work. While the system remained stable for higher sampling times, the output

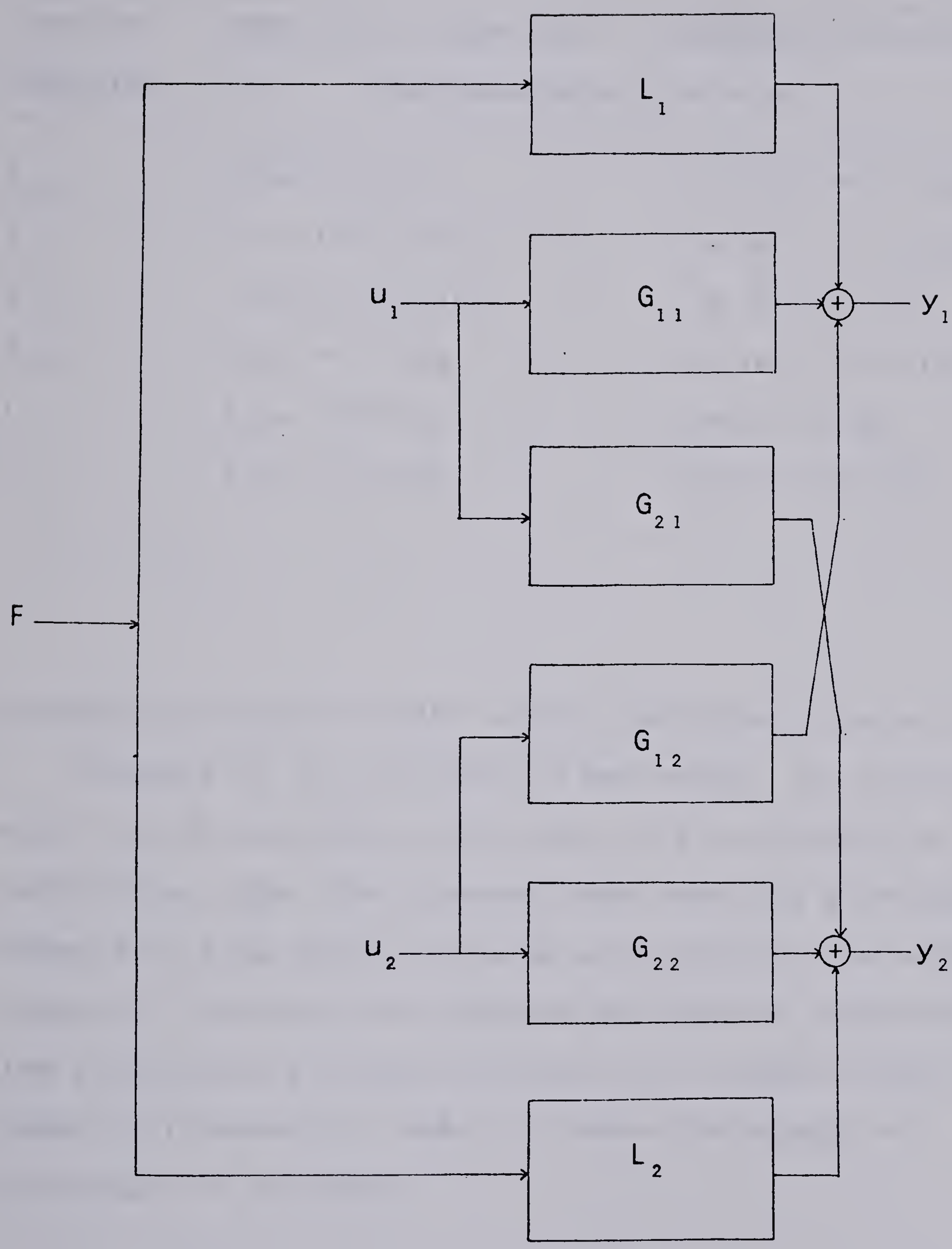


Figure 8: Transfer function model for a binary distillation column [18].

Table VI: Asymmetrical transfer function model of a binary distillation column [18].

TRANSFER FUNCTION	INCREASE IN FLOW RATE	DECREASE IN FLOW RATE
	Time constants in minutes	
G_{11}	$11.8e^{-s}/1+17s$	$14.7e^{-1.4s}/1+15s$
G_{21}	$-17.3e^{-3.5s}/1+20s$	$-19.1e^{-4.1s}/1+22s$
G_{12}	$7.2e^{-6.5s}/1+10s$	$8.1e^{-7s}/1+11s$
G_{22}	$-18.9e^{-3s}/1+13s$	$-19.7e^{-3.9s}/1+12s$
L_1	$4.5e^{-5s}/1+25s$	$5.9e^{-8s}/1+13s$
L_2	$4.3e^{-4s}/1+22s$	$5.2e^{-5.7s}/1+10s$

variance increased and the control performance deteriorated.

Because of the fractional time delays, the discrete model has 62 parameters which have to be estimated by the self-tuning algorithm. However, when sampling a process, the detectable time delay is always a multiple of the sampling interval. Therefore, an adequate description is provided if the time delays are approximated by an integral number of sampling intervals in order to reduce the number of parameters in the model.

5.2 Simulation results

In all simulations, output noise of magnitude $0.01\underline{\underline{I}}$ is included in order to make the results more realistic.

First, the advantages of using feedforward control will be demonstrated. Figures 9a and 9b show the output and input behaviour under multivariable self-tuning control with feedforward control. The sampling time is 7 minutes and all time delays are considered to be zero. The discrete model is given by the following equation:

$$\underline{y}(t+1) =$$

$$\begin{vmatrix} 1.9 & 0.0 \\ 0.0 & 2.0 \end{vmatrix} \underline{y}(t) + \begin{vmatrix} -1.2 & 0.0 \\ 0.0 & -1.3 \end{vmatrix} \underline{y}(t-1) + \begin{vmatrix} 0.2 & 0.0 \\ 0.0 & 0.3 \end{vmatrix} \underline{y}(t-2) +$$

$$\begin{vmatrix} 4.0 & 3.6 \\ -5.1 & -7.9 \end{vmatrix} \underline{u}(t) + \begin{vmatrix} -5.0 & -5.1 \\ 6.7 & 11.3 \end{vmatrix} \underline{u}(t-1) + \begin{vmatrix} 1.5 & 1.8 \\ -2.2 & -4.0 \end{vmatrix} \underline{u}(t-2) +$$

$$\begin{vmatrix} 1.1 \\ 1.2 \end{vmatrix} z(t) + \begin{vmatrix} -1.3 \\ -1.5 \end{vmatrix} z(t-1) + \begin{vmatrix} 0.4 \\ 0.6 \end{vmatrix} z(t-2)$$

After 490 minutes a 10% feed disturbance enters the column. After 1050 minutes the set point of the top composition is increased by 1%. As expected, this change is reflected in the bottom composition because of the significant interaction effect of the reflux flow. After 1610 minutes the set point of the bottom composition is increased by 1%. The top composition is not visibly disturbed although the reflux flow is slightly adjusted. The change in steam flow is almost the same as for the top increase. This is because the gain of the interaction transfer function G_{21} is almost the same as the gain of the direct transfer function G_{22}

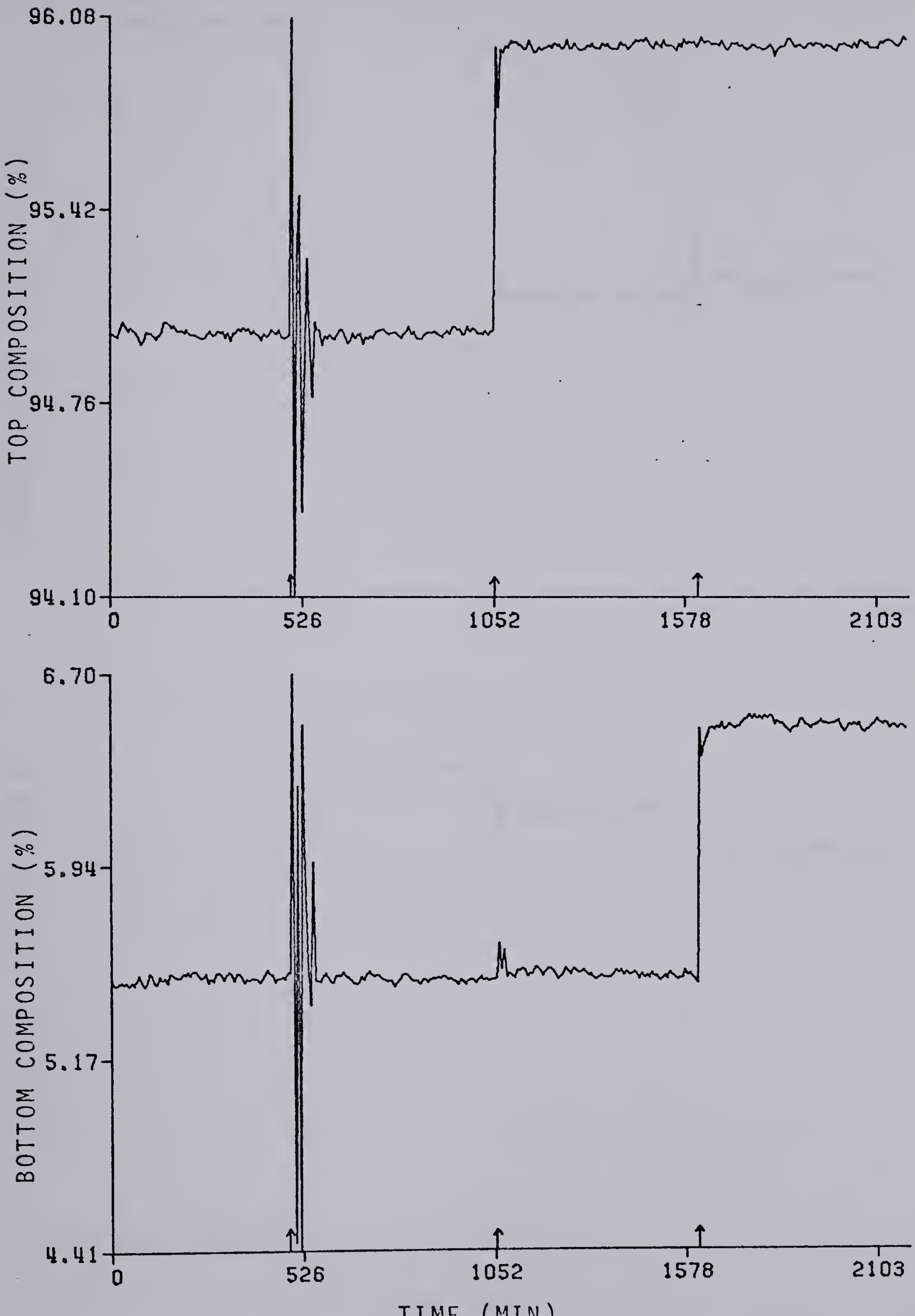


Figure 9a: Simulation of binary distillation column control behaviour. Output behaviour for step increases of 10% in the feed flow rate and 1% in top and bottom composition set points.

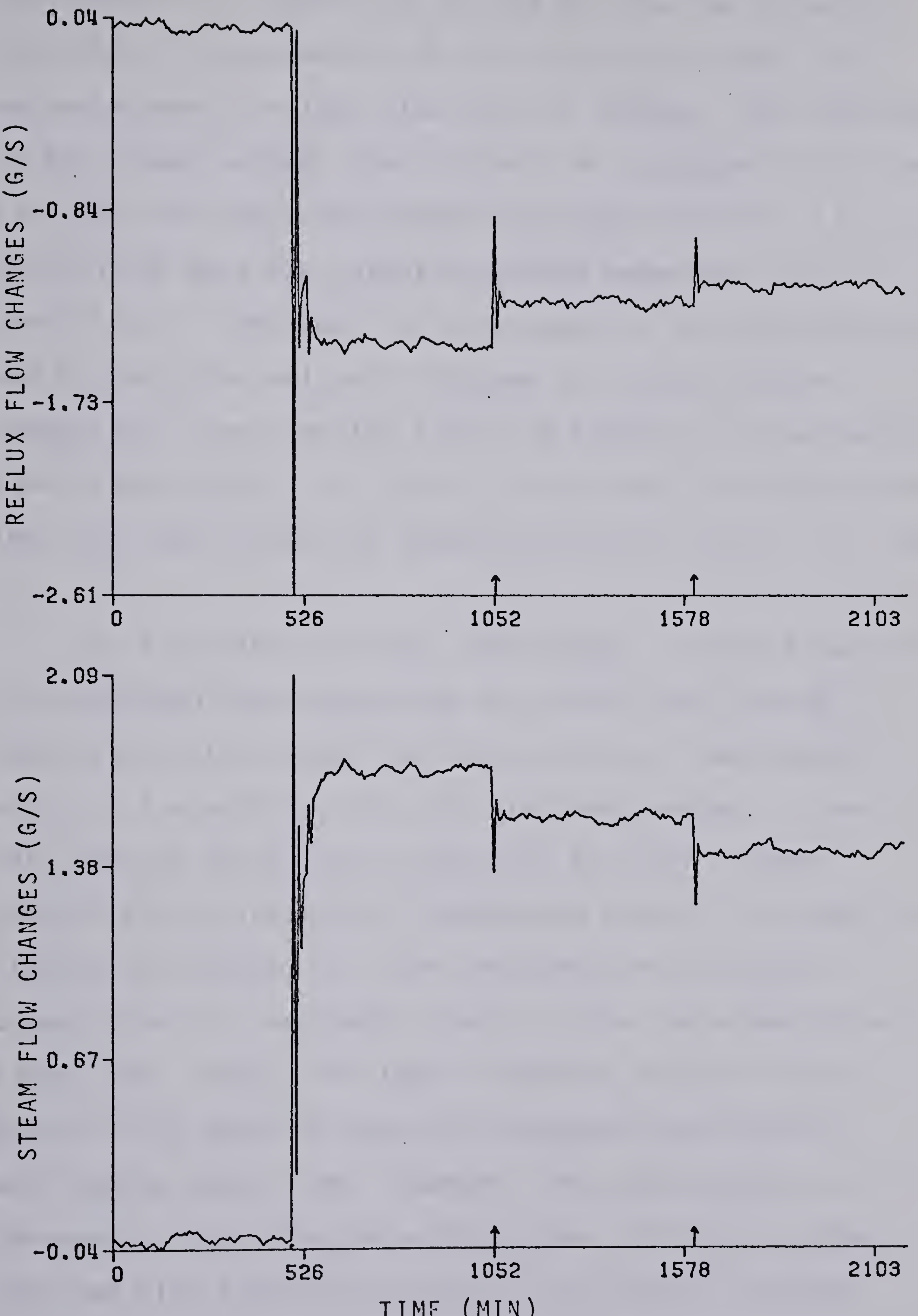


Figure 9b: Simulation of binary distillation column control behaviour. Input behaviour for step increases of 10% in the feed flow rate and 1% in top and bottom composition set points.

(see Table VI). Figures 9c, 9d and 9e show the parameter estimates. The parameters of the disturbance model are estimated when the feed flow rate is changed. They converge to the actual values. None of the α or β parameters converge fast, but this does not prevent very good control. Figures 10a and 10b give the output and input behaviour for the simulation of the model for a decrease in the feed flow rate and for negative set point changes in top and bottom composition. The sampling time is 5 minutes. The estimated time delays for G_{11}, G_{21} and G_{22} are taken to be zero, and they are taken to be one sampling interval for G_{12}, L_1 and L_2.

It is possible to treat the changes in feed flow rate as an unknown disturbance and to use the self-tuning controller with integral action instead of feedforward control. A simulation run with the same changes in feed flow rate and set points as in Figure 9, but with integral control action instead of feedforward control, is shown in Figures 11a through 11e. The influence on the bottom composition of a set point change in the top composition is almost zero. Thus, this type of control action is more suitable for handling set point changes than ordinary self-tuning controllers. However, the output behaviour in the case of feed flow rate changes was inferior to that obtained with feedforward control; the output variance increased by a factor ten. In a practical application, both measurable and unmeasurable disturbances, and set point

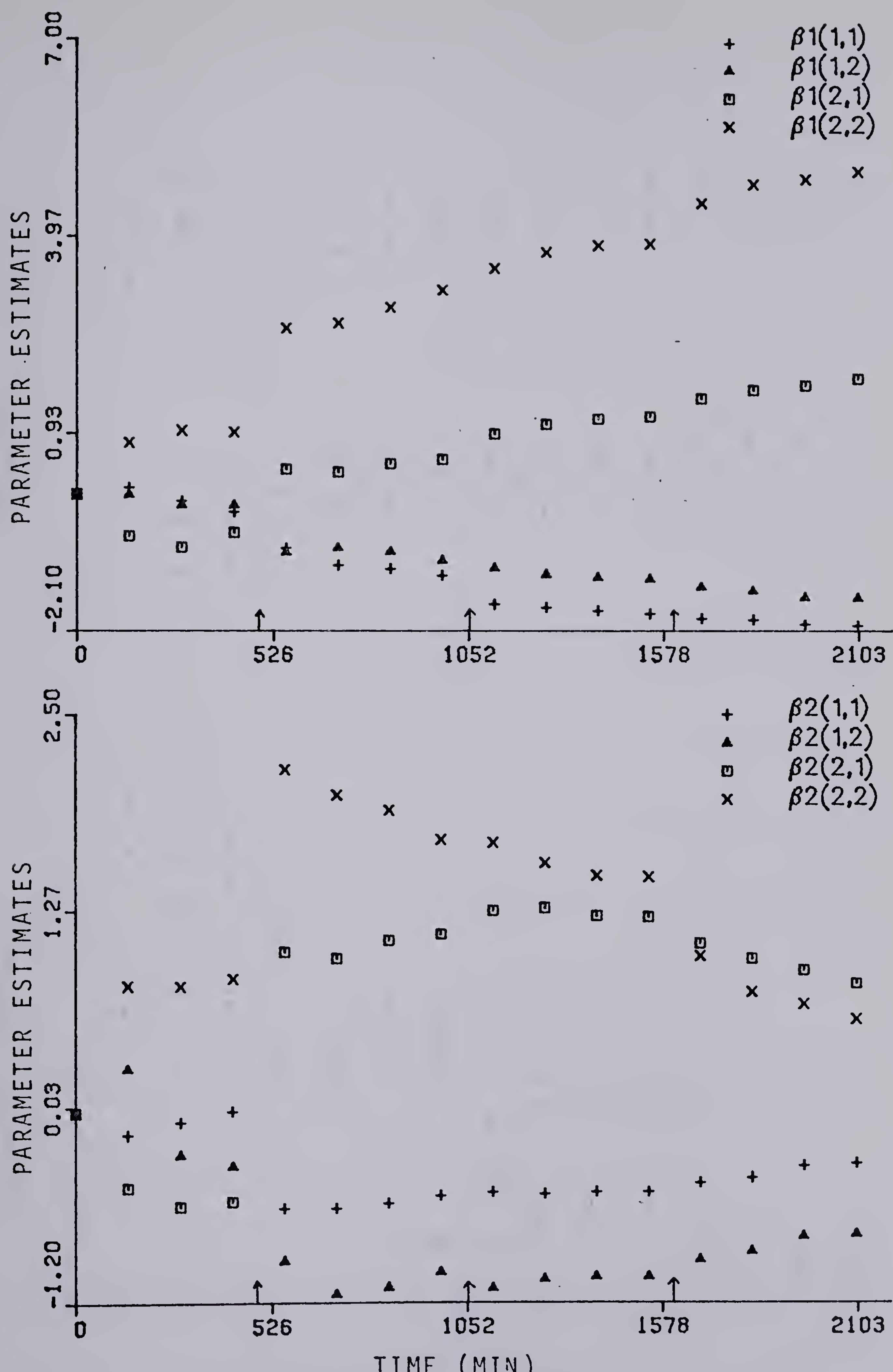


Figure 9c: Simulation of binary distillation column control behaviour. Parameter estimates for step increases of 10% in the feed flow rate and 1% in top and bottom composition set point.

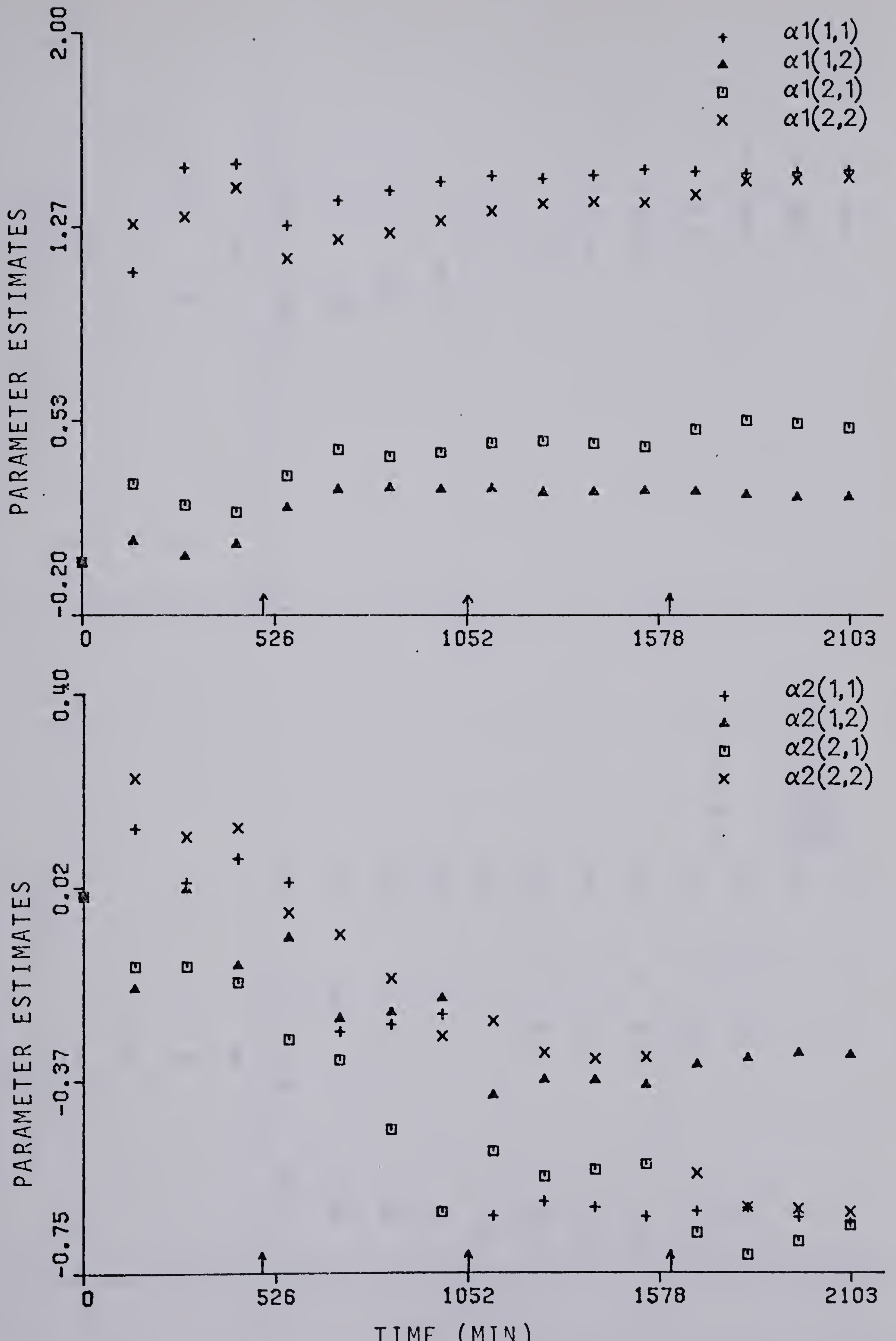


Figure 9d: Simulation of binary distillation column. Parameter estimates for step increases of 10% in the feed flow rate and 1% in top and bottom composition set point.

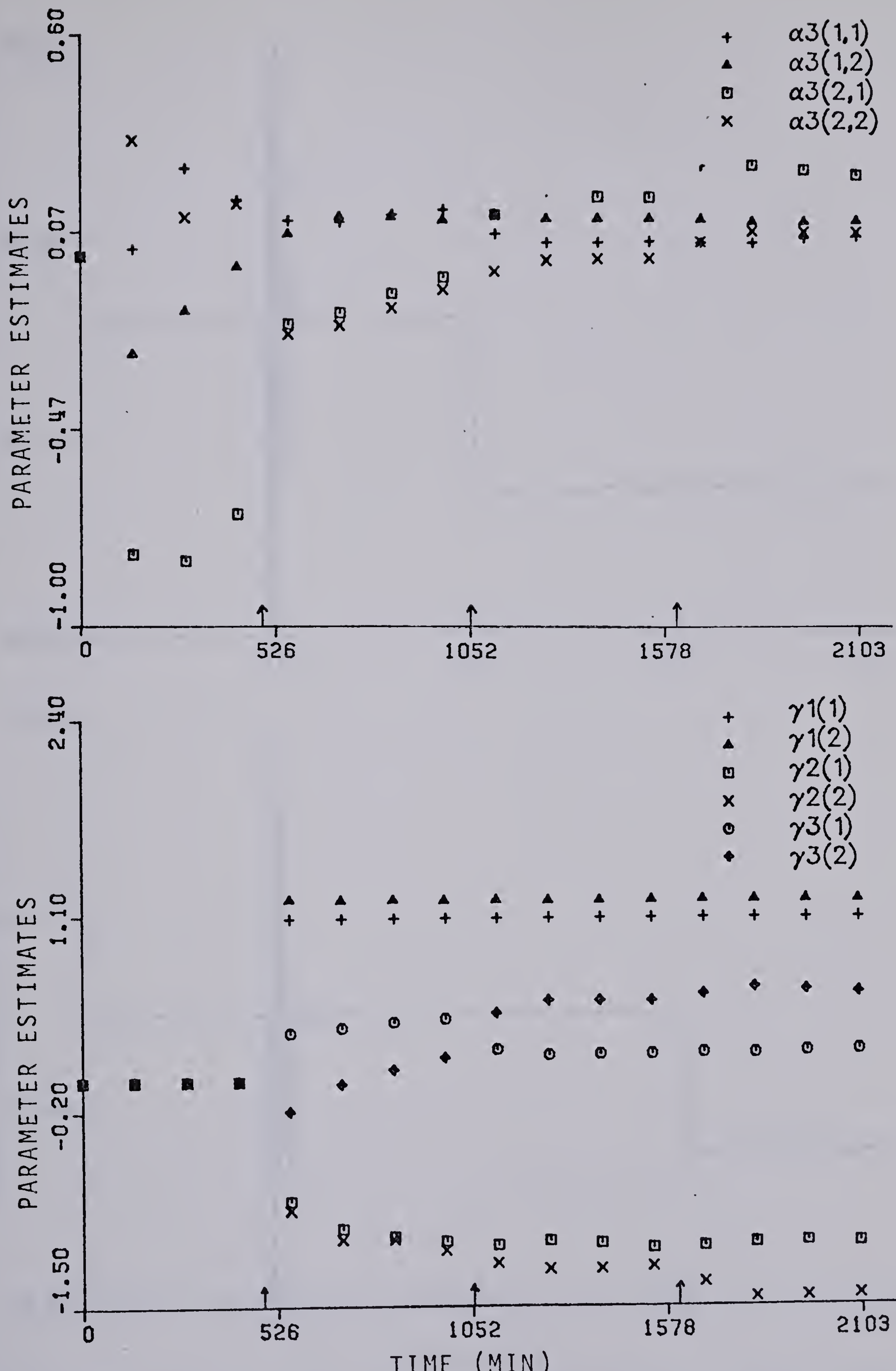


Figure 9e: Simulation of binary distillation column control behaviour. Parameter estimates for step increases of 10% in the feed flow rate and 1% in top and bottom composition set point.

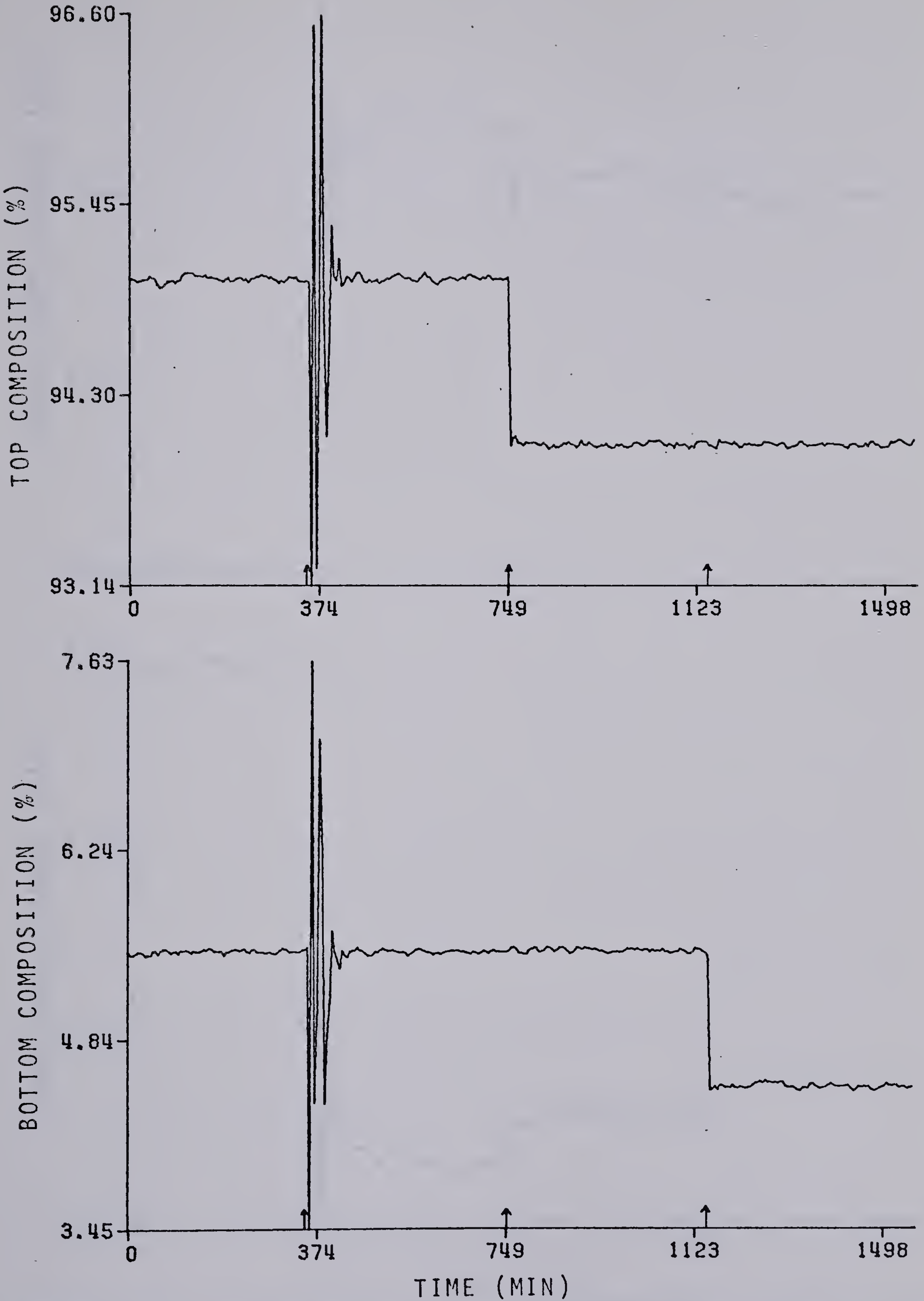


Figure 10a: Simulation of binary distillation column control behaviour. Output behaviour for step decreases of 10% in the feed flow rate and 1% in top and bottom composition set points.

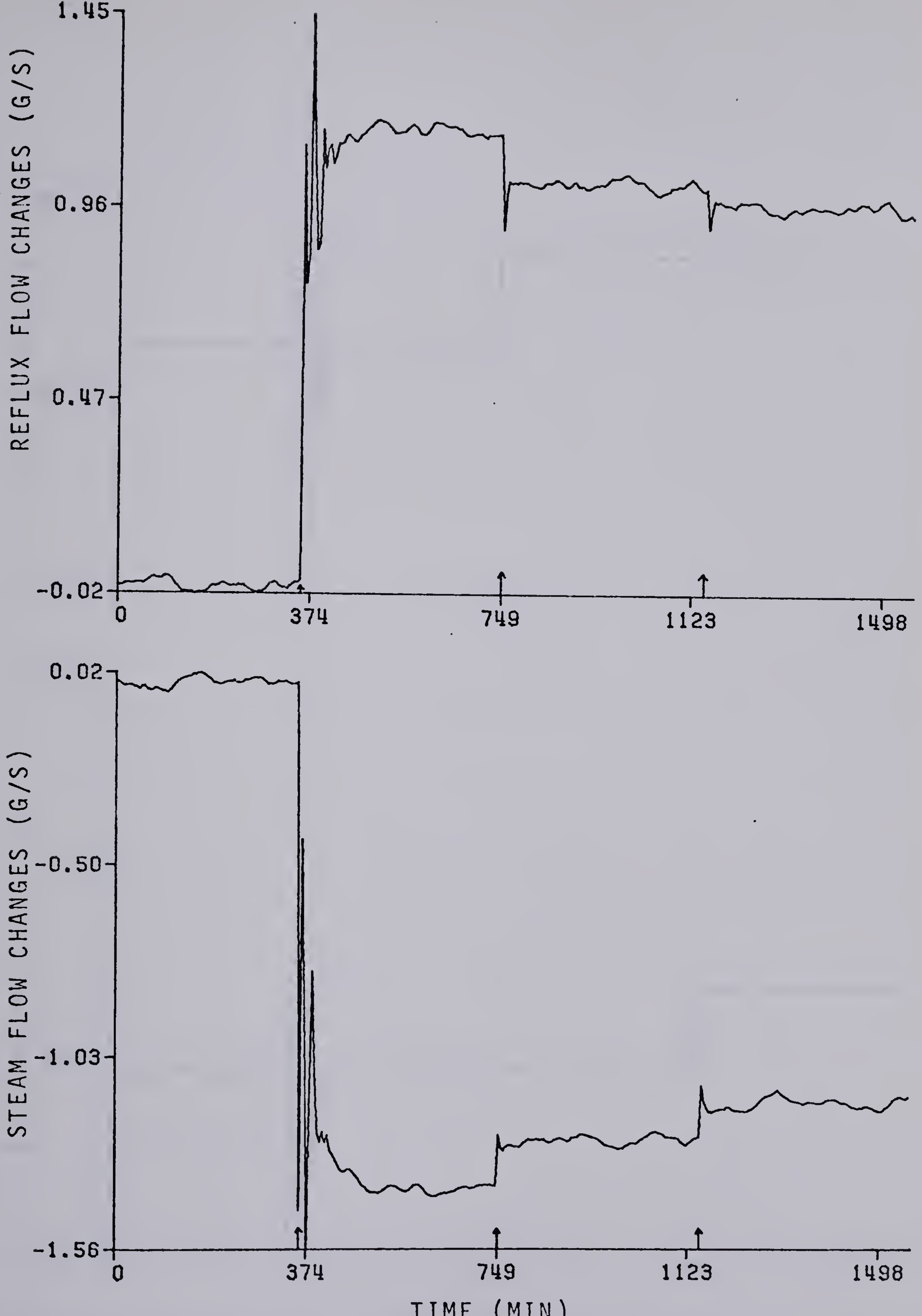


Figure 10b: Simulation of binary distillation column control behaviour. Input behaviour for step decreases of 10% in the feed flow rate and 1% in the top and bottom composition set points.

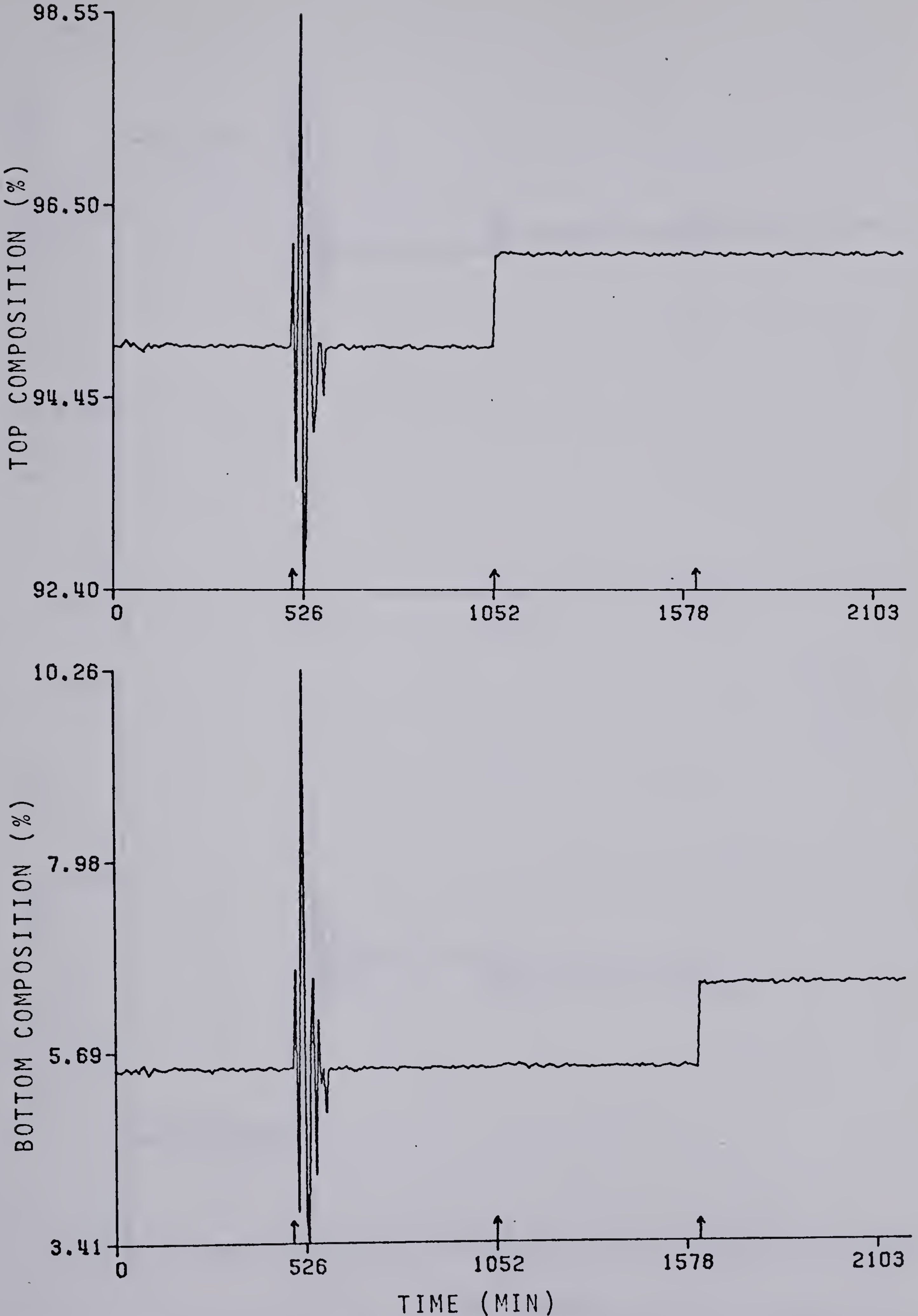


Figure 11a: Simulation of binary distillation column control behaviour using integral control action. Output signals for a 10% increase in feed flow rate and 1% in top and bottom composition set point.

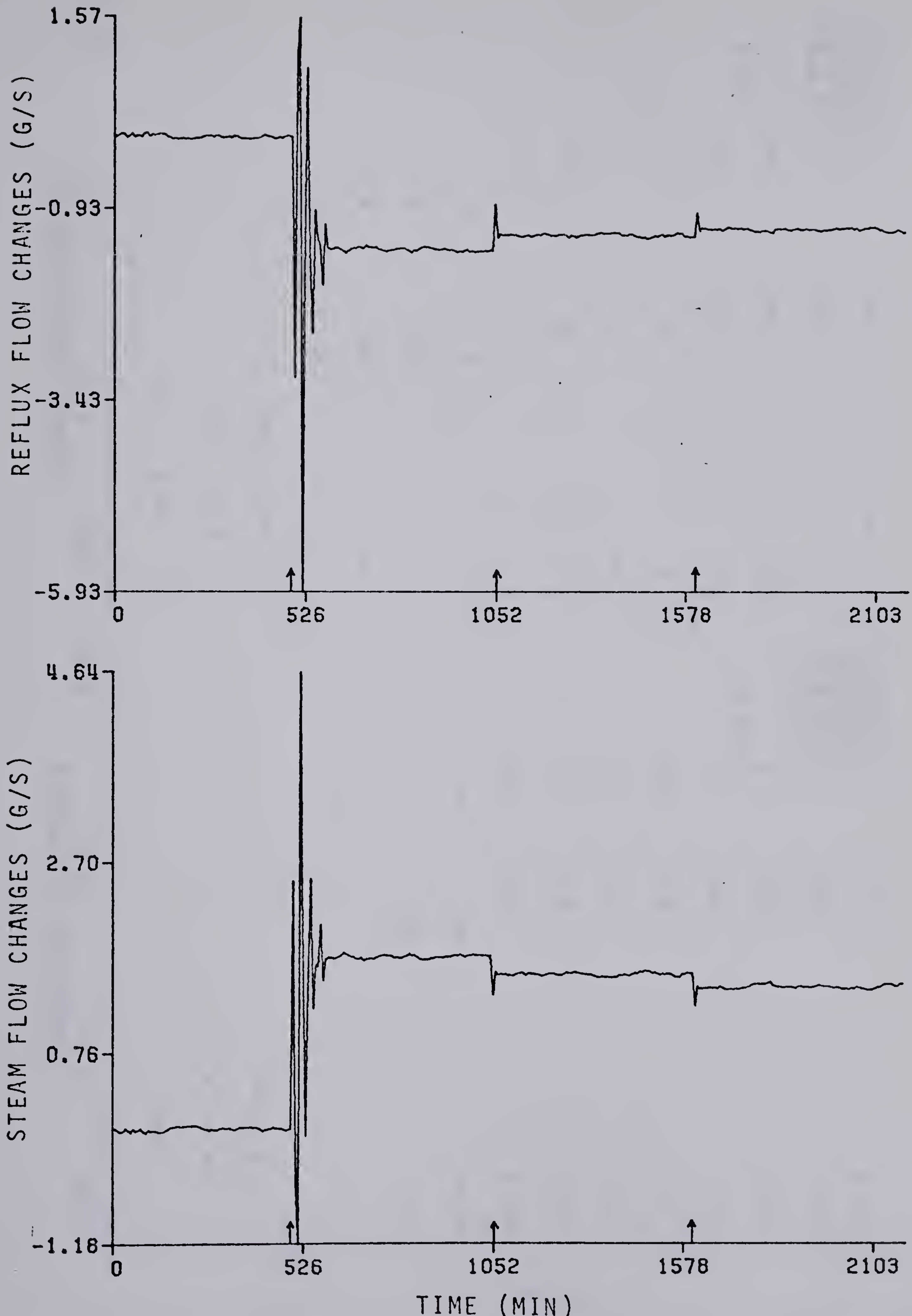


Figure 11b: Simulation of binary distillation column control behaviour using integral control action. Input signals for a 10% increase in feed flow rate and 1% in top and bottom composition set point.

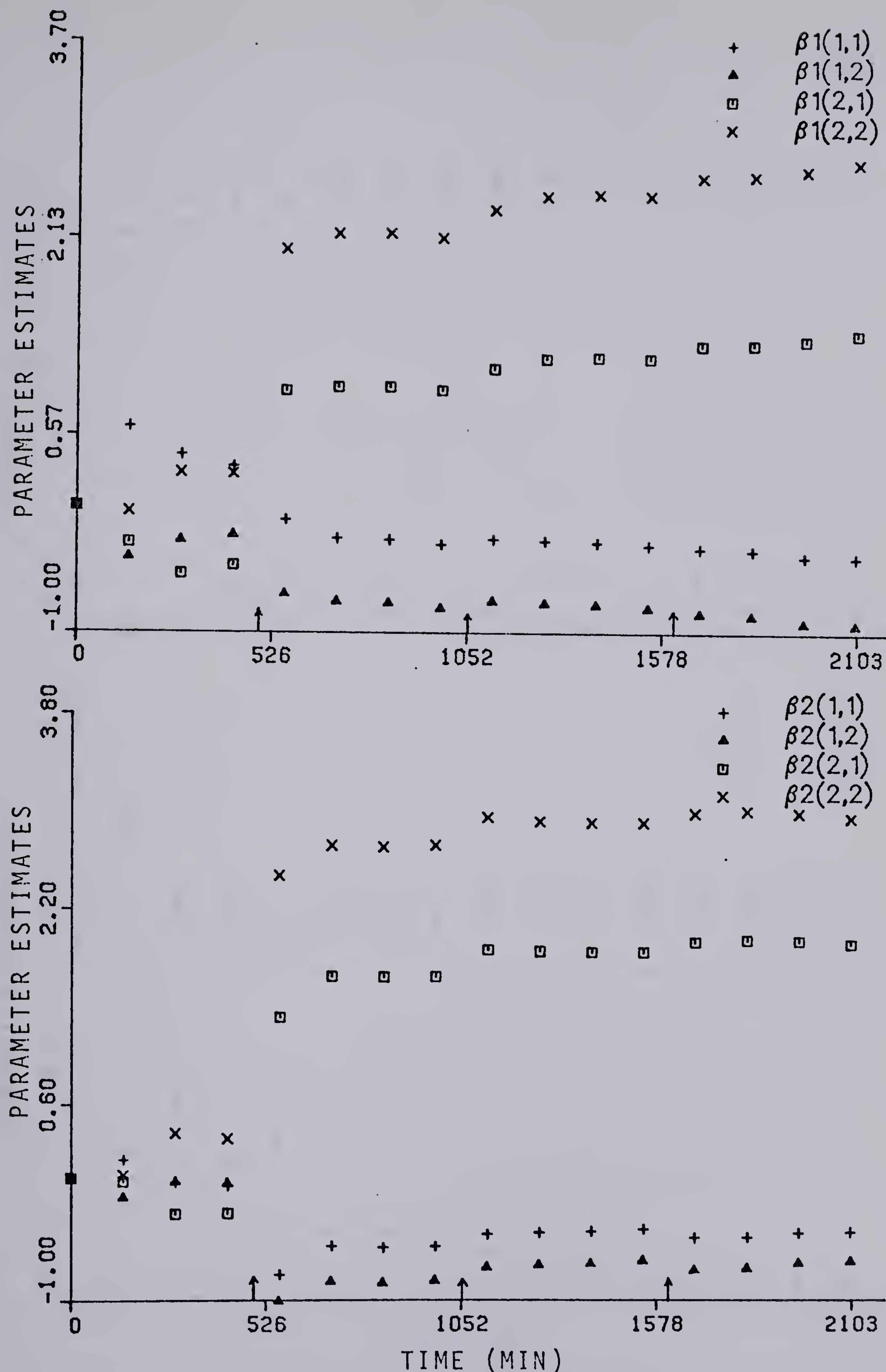
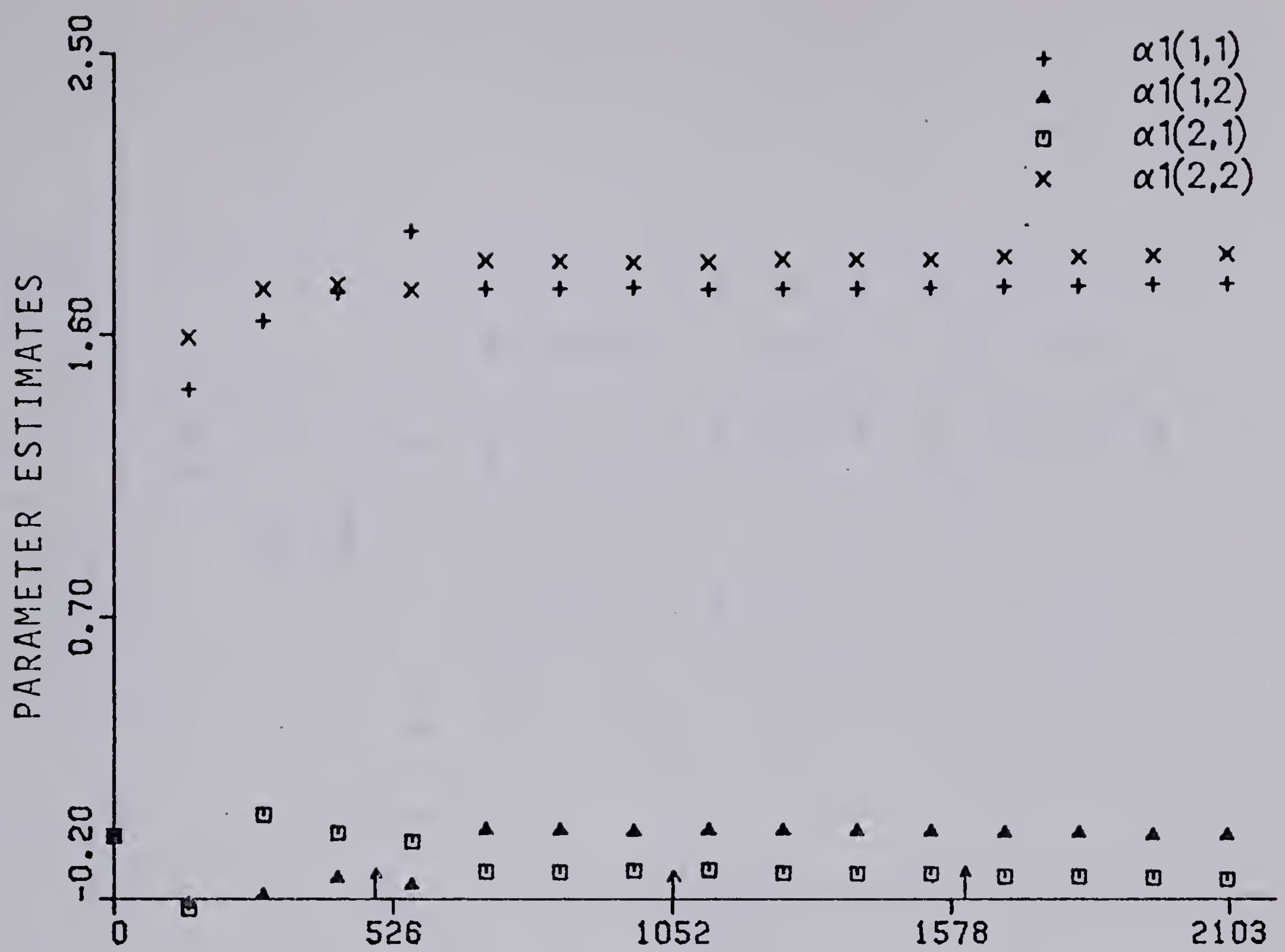


Figure 11c: Simulation of binary distillation column control behaviour using integral control action: parameters for a 10% increase in feed flow rate and 1% in top and bottom composition set point.



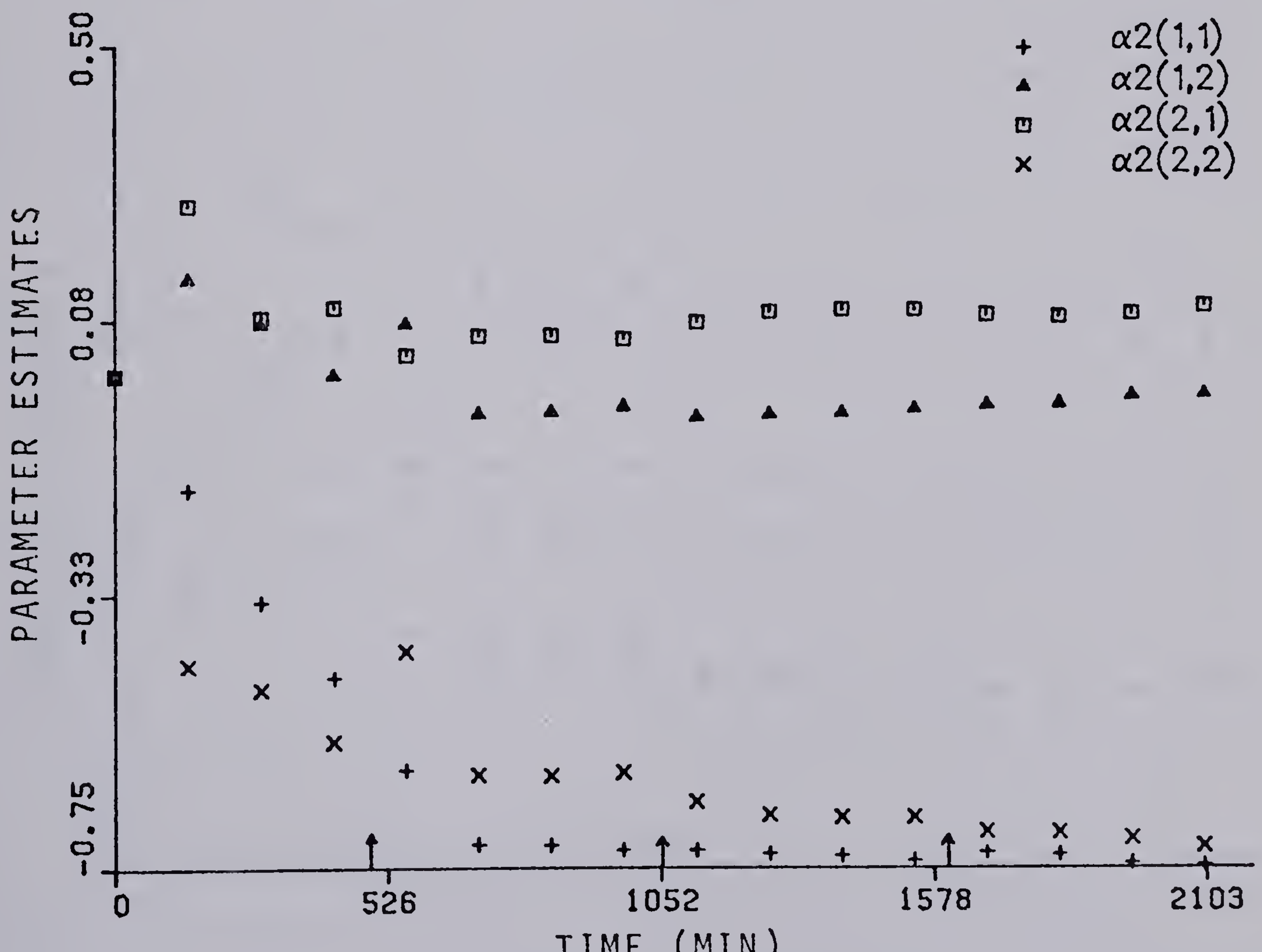


Figure 11d: Simulation of binary distillation column control behaviour using integral control action: parameters for a 10% increase in feed flow rate and 1% in top and bottom composition set point.

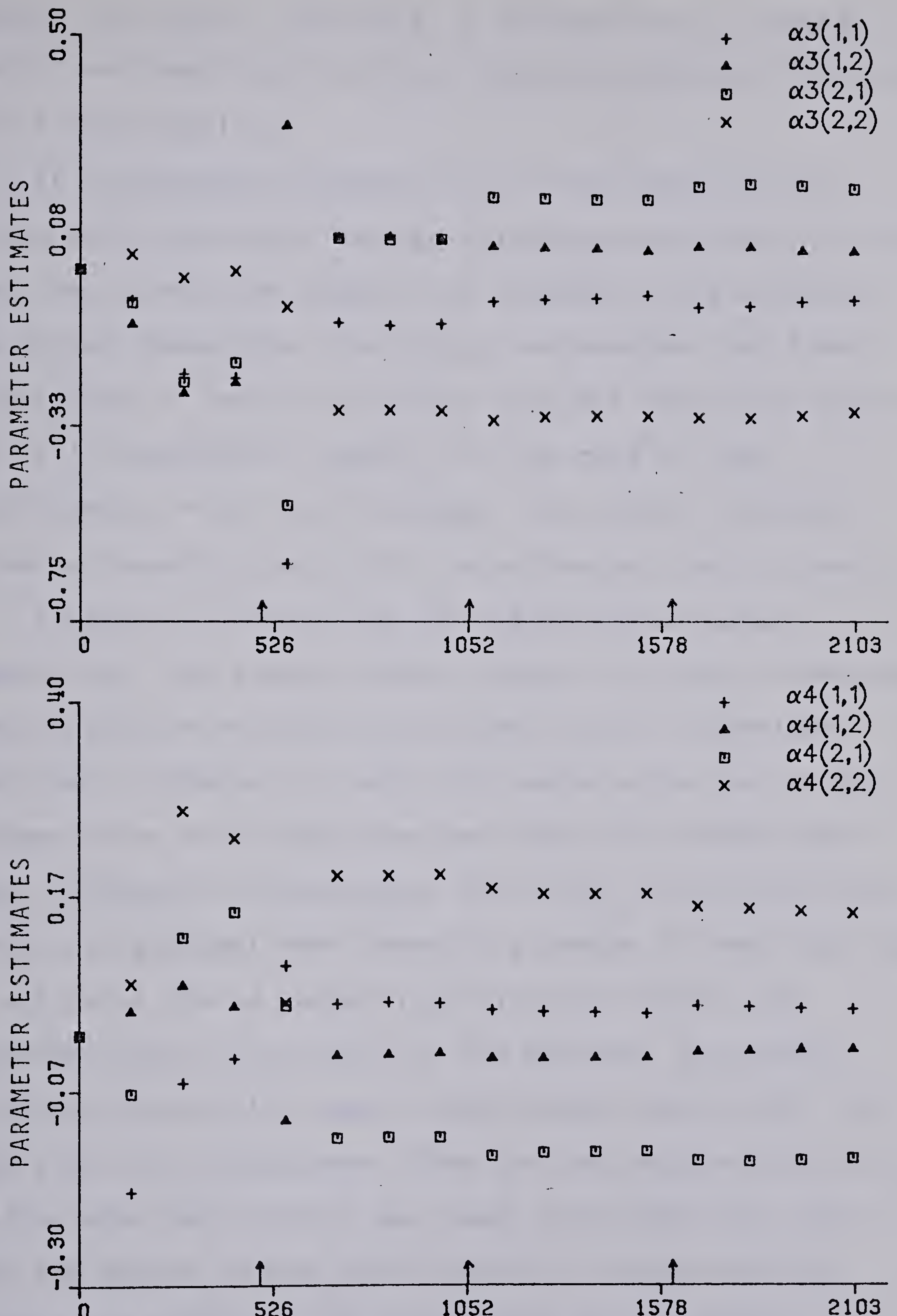


Figure 11e: Simulation of binary distillation column control behaviour using integral control action: parameters for a 10% increase in feed flow rate and 1% in top and bottom composition set point.

changes, may occur. Therefore, a combination of integral control and feedforward control should be added to the basic self-tuning regulator.

It is possible to control the binary distillation column with two single loop self-tuning controllers. In this case the interactive signals are treated as disturbances. Simulation showed that the output variance was ten times larger than in the multivariable case but regulatory control was still acceptable. However, in the case of feed disturbances or set point changes, the output variance became unreasonably high and the system went out of control.

Instead of controlling the instantaneous output composition, the average output composition over a specified time period can be controlled using the RAFT technique described in Chapter 3. Table VII compares the sum of the average error over seven time periods of 50 minutes each, under different circumstances. The effect of controlling the average is greatest when there is a change in feed flow rate or set point. For a large signal-to-noise ratio, the technique has little effect on the average. The output variance remained the same in both cases except after the feed flow rate disturbance. Then the variance was actually smaller when RAFT control was used. The reason for this is that the change in the input signal is larger and this reduces the effect of the disturbance on the output.

Table VII: Comparison of output variance and average output error over seven time periods of 50 minutes each, for binary distillation column control.

CHANGE	AVERAGE ERROR Y_1		VARIANCE Y_1	
	STR	STR-RAFT	STR	STR-RAFT
NONE	0.0467	0.0063	0.033	0.053
FEED FLOW	0.3562	0.0059	13.92	6.47
SET POINT Y_1	0.2157	0.0052	2.05	2.21
SET POINT Y_2	0.0230	0.0035	0.025	0.028

6. SIMULATION OF BINARY DISTILLATION COLUMN CONTROL BEHAVIOUR

6.1 Introduction

In this chapter, a simulation study of a pilot scale binary distillation column controlled by a multivariable self-tuning controller is described. The column has a diameter of 22.86 cm and contains 8 bubble cap trays spaced at 30.48 cm with the methanol-water feed introduced at the fourth stage. The column which is equipped with a thermosyphon reboiler and a total condenser, is shown schematically in Figure 12. A detailed description of the column and its associated instrumentation and equipment is given by Pacey [19] and Svrcek [21].

Several models of the column have been presented for simulation of the column's dynamic behaviour and for the determination of suitable control laws. Their complexity ranges from a nonlinear differential equation model in 20 variables to a linear first order plus time delay transfer function model with two inputs and two outputs. While the more complex models give a better prediction of the dynamic behaviour of the process, they are too difficult to use when deriving a control law with fixed parameters. The model used in this study, which is based an mass, composition and energy balance relationships for each tray, condenser and reboiler, was established by Kan [10]. The controlled variables are the compositions of the top and bottom

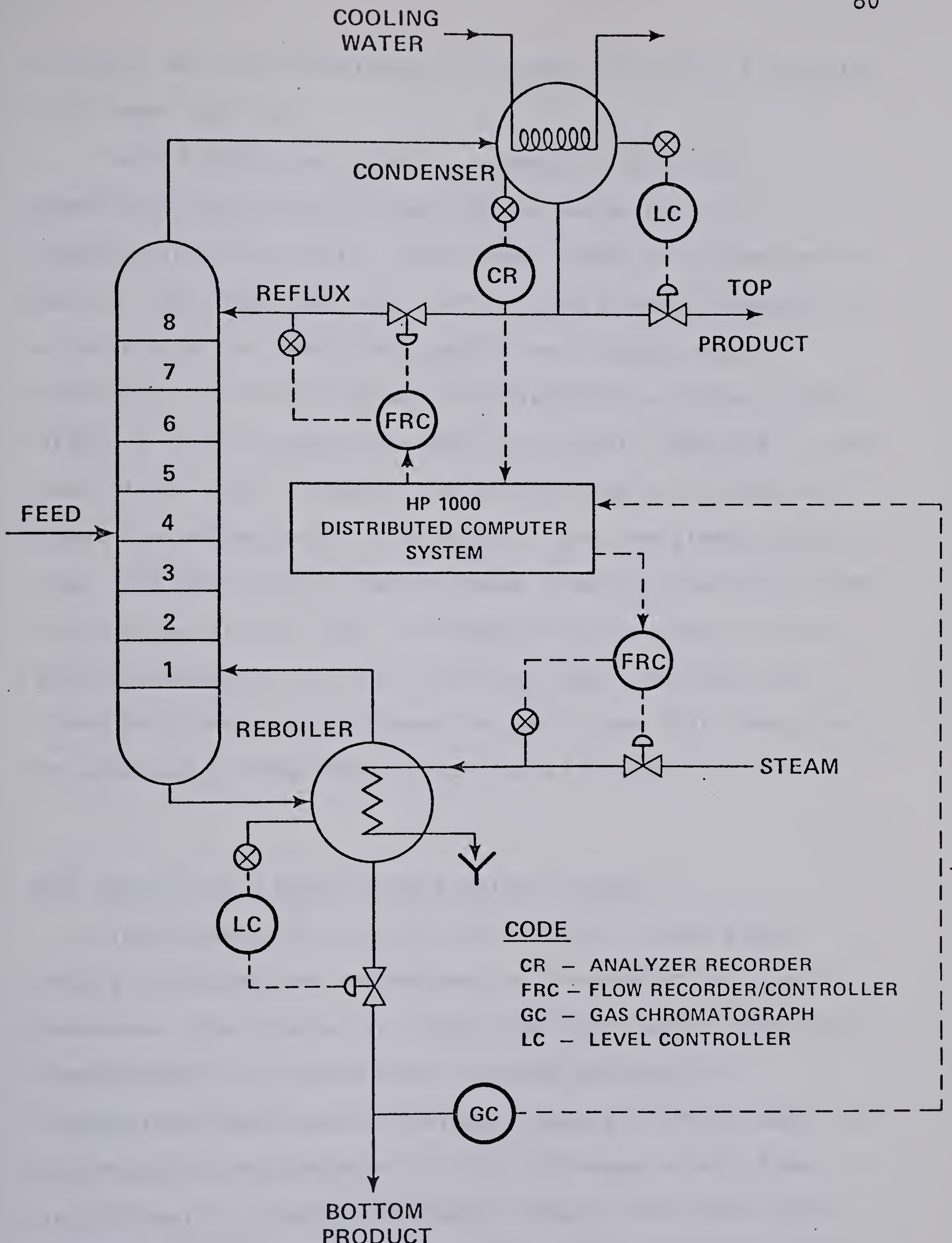


Figure 12: Schematic diagram of the distillation column.

products and the manipulated variables are reflux flow rate and steam flow rate.

Self-tuning controllers can adapt to different operating conditions by changing the parameters of a prediction model on-line. This model, used in the estimation part of the algorithm, must reflect the process dynamics for a certain set of operating conditions. Because most processes are linear for small perturbations around steady state, a linear prediction model is usually adequate. In the case of set point changes, the self-tuning controller will adapt the parameters in the model to the new steady state. Thus, the model can be much simpler than one characterizing the process at all times. Although the parameters of the prediction model can vary, the order and time delay are fixed and they must be chosen in such a way that they give an adequate process description for all times.

6.2 Control of a binary distillation column

The control of a binary distillation column poses several problems not encountered in the control of simpler processes. The process is highly nonlinear which means that the parameters of conventional analog and digital controllers need constant tuning. Because a linear model can not describe the dynamics of such a process at all times, it is difficult to predict the output and to find the correct parameters for the controllers. One set of parameters will not be adequate for all process states in the case of severe

nonlinearities.

The top and bottom compositions respond slowly to disturbances in the feed. This means that the internal system undergoes large changes before control action can be taken and the compositions reflect the disturbance over a long period of time. There is also interaction between the top and bottom composition control loops, so tuning of one loop will affect the other. This makes it almost impossible to design a controller for one loop without consideration of the other loop.

Several control schemes have been tested on simulations of the distillation column mentioned above, and on the column itself. At first, only the overhead composition was computer controlled, while the bottom loop remained under manual control [19]. The control schemes employed a two mode feedback controller with and without feedforward compensation. Later, simultaneous control of both bottom and top composition was implemented. Most schemes tried to reduce interaction by adding decoupling elements to the controllers [13]. All of these controllers were based on a transfer function model of the column. Several multivariable control schemes were implemented [16], but in some cases the resulting control was worse than for multiloop control.

The nonlinearity of this process suggested the implementation of self-tuning regulators. This would make it unnecessary to design a model which is both complex enough to describe the dynamics of the column, and at the same time

simple enough to allow the calculation of the parameters of a controller.

Sastry et al [20] used a self-tuning regulator to control the top composition using reflux flow rate as the manipulated variable, while the bottom loop was not controlled as the steam flow rate was kept constant. Self-tuning regulators were compared with PI controllers for set point changes and feed flow disturbances, however, these changes were introduced at the exact time the STR algorithm was started with all parameters still zero. This is not realistic because the response of the closed loop system to these changes is affected by the start-up behaviour of the algorithm. Still, the experimental results proved the superiority of the self-tuning regulator over a well tuned PI controller in handling feed flow disturbances. The main reason for this is that the self-tuning regulator can adapt itself to the nonlinear behaviour of the column while the PI controller can not.

In a subsequent thesis [13], two single loop self-tuning regulators of the Clarke and Gawthrop type were used to simultaneously control the top and bottom compositions of the column, however this scheme has the disadvantage that it does not account for the interaction in the distillation column. The multivariable self-tuning controller described in this study attempts to deal with both nonlinearities and interaction in the process, two of the main difficulties encountered in distillation column

control.

6.3 Simulation results

6.3.1 Selection of the prediction model

The time constants of the pilot scale column are significantly smaller than those of an industrial column so consequently the sampling rate for digital control must be higher. This poses a practical problem because the sample analyzer imposes a large time delay in the bottom loop. In single loop control, this time delay should not be to much of a problem because it can be taken into account when choosing the sampling time and determining the control parameters. However, for the self-tuning controller described in this thesis, this large time delay presents a design problem, because the sampling rate for the top and bottom loops must be the same and furthermore the assumed time delay in the prediction model must be identical for both loops. If this predicted delay were not the same, the leading $\underline{\underline{B}}$ matrices in the prediction model would be of the form:

$$\underline{\underline{B}}i = \begin{vmatrix} 0.0 & 0.0 \\ a & b \end{vmatrix}$$

and if these matrices are estimated correctly by the self-tuning algorithm $\underline{\underline{\beta 0}}$ would also be singular and it would be impossible to calculate its inverse for use in the

control algorithm. To overcome this difficulty, the time delay for the bottom loop was assumed to be of the same order of magnitude as the one for the top loop. Although this is not true for the pilot scale column, it is probably a reasonable assumption for a large scale system. (The effect of large time delays in one loop was investigated in later simulations.)

Because of the nonlinearity of the column, it is also difficult to find a good estimate for the leading $\underline{\underline{B}}$ matrix, and as a result it was almost impossible to fix this value in initial runs. All attempts in this direction failed. After a reasonable value had been obtained in the first simulation run, $\underline{\underline{\beta}}0$ could be fixed, but this would have prevented this matrix from changing when set point changes or disturbances occurred. In general, it was judged better to include the estimation of $\underline{\underline{B}}0$ in the algorithm at all times.

Before starting the simulations, a prediction model must be chosen. Because previous workers obtained reasonable results when using first- and second-order models to simulate the column behaviour for fixed set points, corresponding pulse transfer functions rewritten as matrix difference equations, were used for the prediction model of the self-tuning regulator. As the column is usually subject to unknown disturbances, integral control action was included in the self-tuning regulator. The corresponding prediction model for a first-order system with normalized

output variables and zero time delay is:

$$\underline{y}(t) = \underline{\underline{A}}1\underline{y}(t-1) + \underline{\underline{A}}2\underline{y}(t-2) + \underline{\underline{B}}0\ \underline{u}(t-1) + \underline{\underline{B}}1\ \underline{u}(t-2)$$

In the case of feed flow changes, the effect of a feedforward compensator was investigated. Noise with a zero mean value and covariance matrix $\sigma\underline{\underline{I}}$ was added to the output signals in order to simulate the column behaviour more realistically.

Output and input behaviour for a sampling time of 64 seconds are shown in Figures 13a and 13b. The estimated time delay has been assumed to be zero and the noise covariance is $0.001\underline{\underline{I}}$. After 80 minutes a feed flow disturbance of -20% is introduced and after 120 minutes the feed flow is reset to its original rate. The disturbance is barely noticeable in the output because of the noise, but the input signals are adapted to the new situation. As can be seen in Figures 13c and 13d, not all parameter estimates have converged after 80 minutes because if they had, their values after 160 minutes would be the same as before the first disturbance was introduced. After 160 minutes only $\alpha 1(2,2)$ and $\alpha 2(2,2)$ are still changing significantly. Despite this, control is very good. Good control performance was also obtained with a sampling time of 32 seconds and when using either 32 or 64 seconds plus a predicted time delay of one.

Table VIII summarizes the resulting output variances for different prediction models for a feed flow disturbance

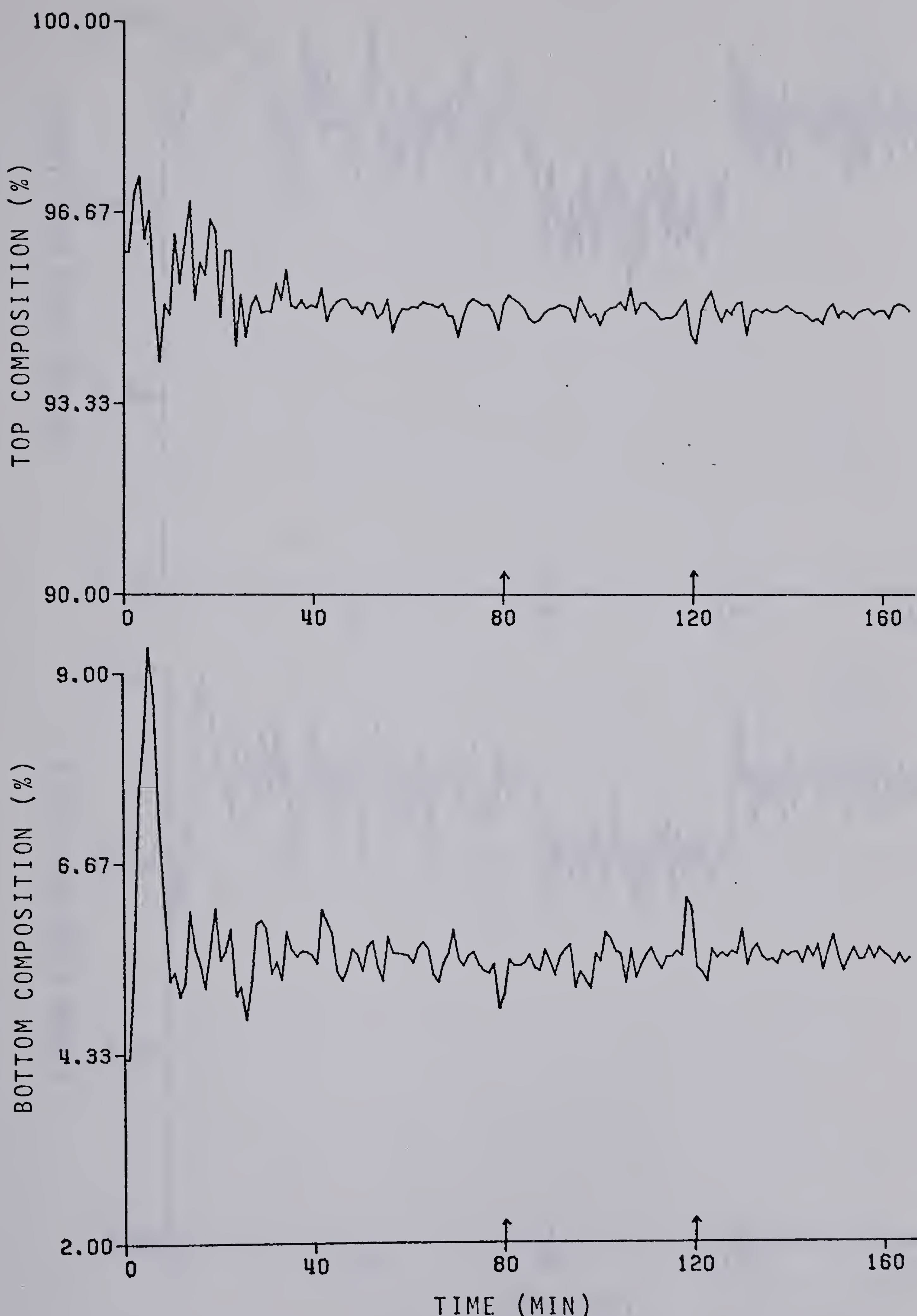


Figure 13a: Simulation of control behaviour: output response for step changes of 20% in the feed flow rate.

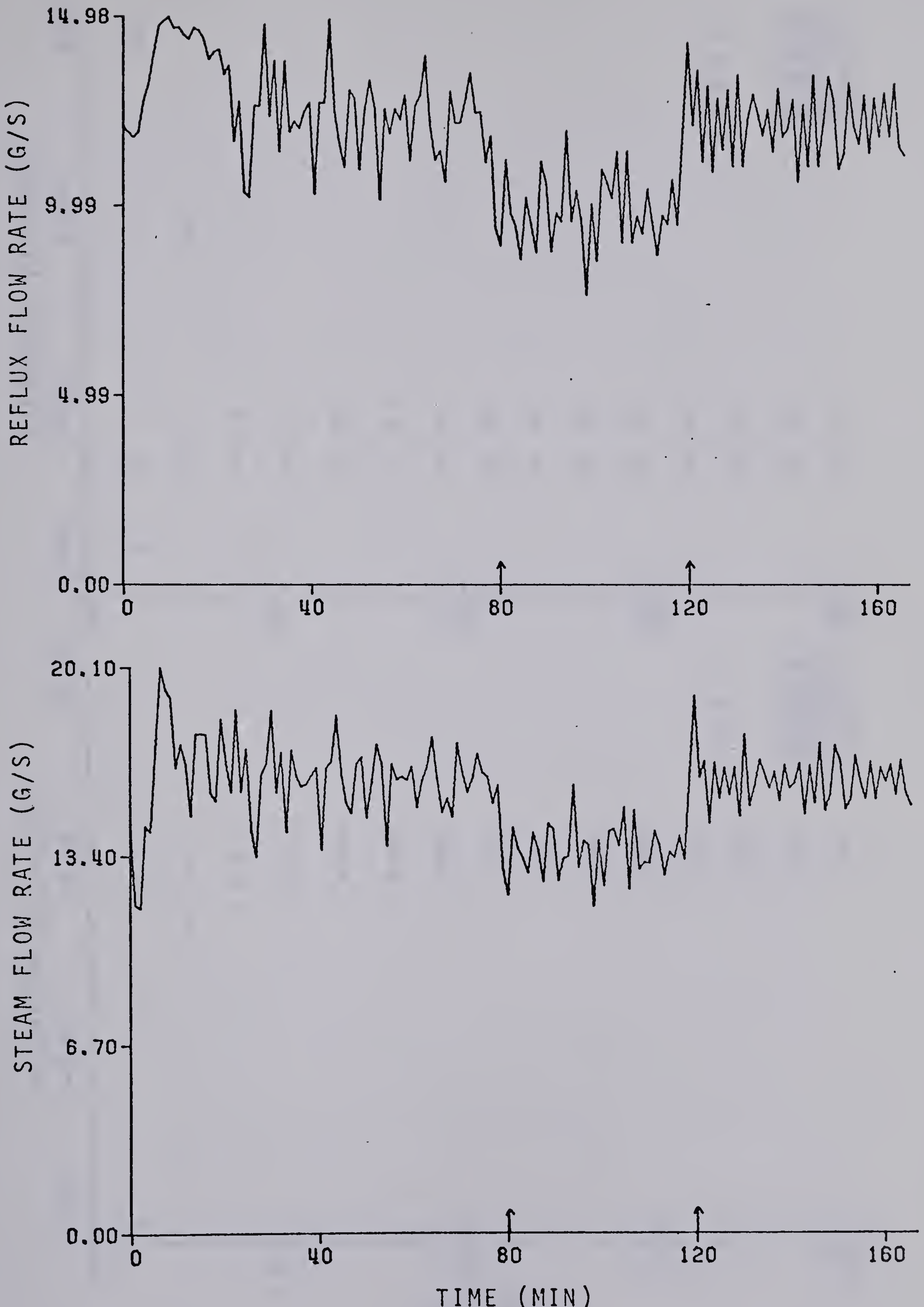


Figure 13b: Simulation of control behaviour: input response for step changes of 20% in the feed flow rate.

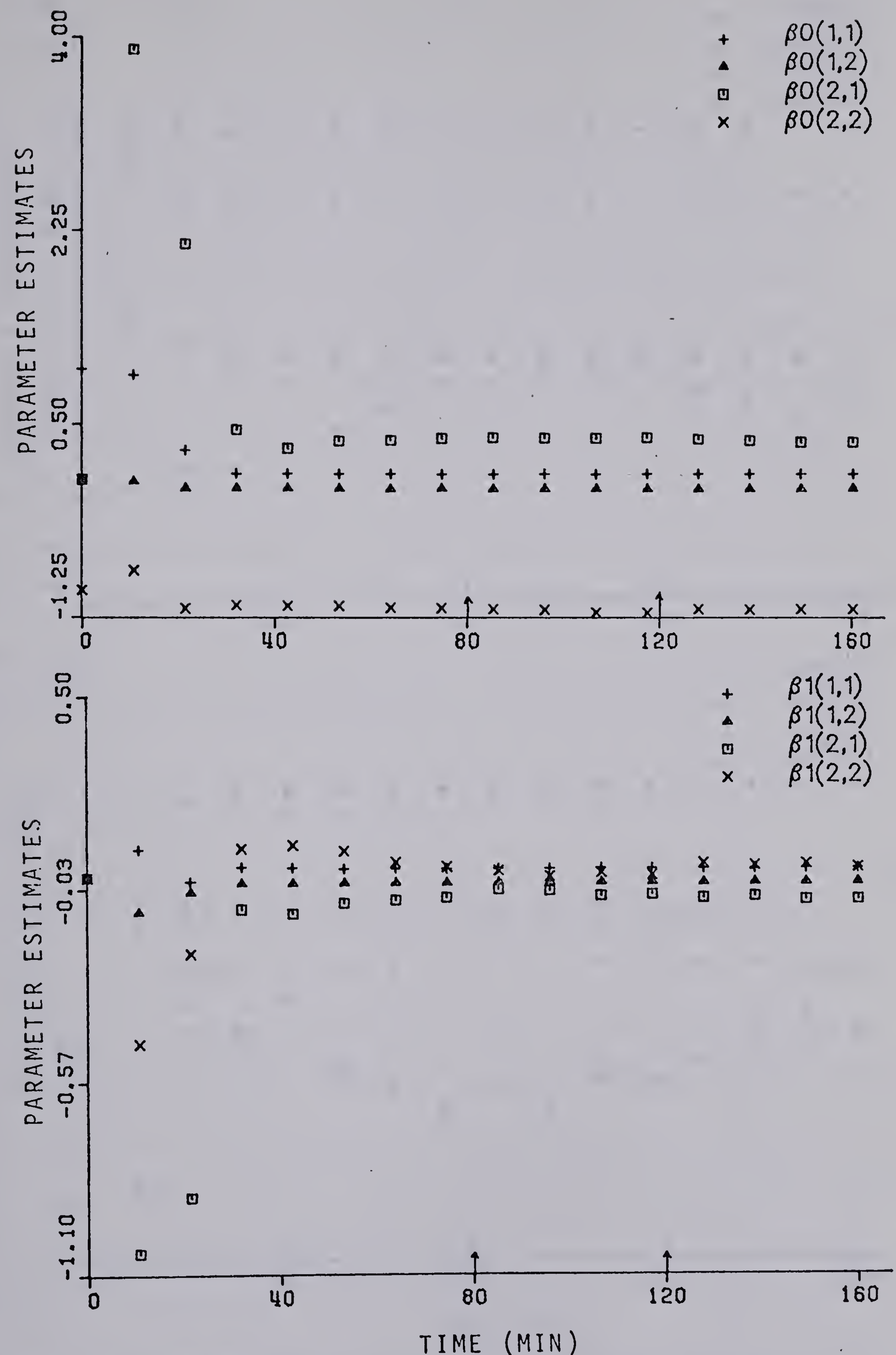


Figure 13c: Simulation of control behaviour: parameter adaption for step changes of 20% in the feed flow rate.

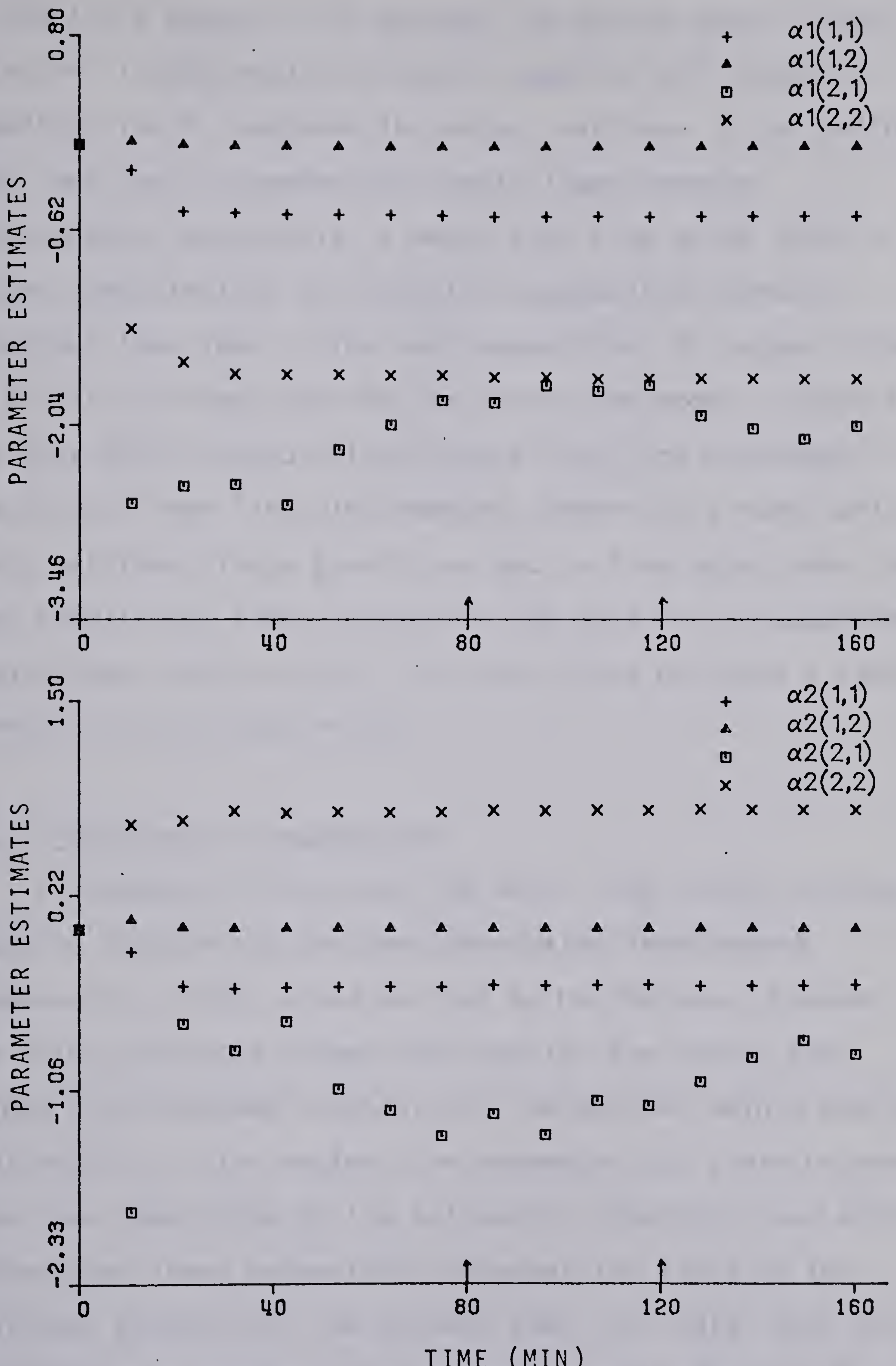


Figure 13d: Simulation of control behaviour: parameter adaption for step changes of 20% in the feed flow rate.

of 30% for a period of 40 minutes. As can be seen, a time delay k=1 in the prediction model together with a small sampling time T_s improves the output variance in the bottom loop, but the top composition control performance deteriorates. Apparently, a model with time delay gives a better approximation of the bottom composition dynamic behaviour than that of the top composition. It proves to be difficult to dimensionalize the prediction model in such a way that both top and bottom compositions are optimized in the case of feed flow disturbances. Generally a model with two $\underline{\underline{A}}$ matrices, three $\underline{\underline{B}}$ matrices and no time delay gave the best results and those values will be used in all subsequent simulations. Unfortunately this model does not have a simple transfer function equivalent.

6.3.2 Feedforward compensation

If changes in the feed flow occur, the output variance would be expected to decrease when using feedforward compensation. This turned out not to be the case, however: the output variance stayed the same for the top of the column and increased slightly for the bottom, mainly due to nonlinearity of the system. The parameters of a disturbance model are identified by the estimation algorithm, and after convergence these parameters represent the state of the nonlinear process for the current feed flow rate. When the feed flow rate changes, the parameters must adapt to the new process state and this usually takes more than one sampling

Table VIII: Accumulated output variance for top and bottom composition after 160 minutes. N and M are the number of estimated $\underline{\underline{A}}$ matrices and $\underline{\underline{B}}$ matrices respectively.

N	M	T_s	k	AIE_1 (160)	AIE_2 (160)
2	2	32	0	0.004969	2.1593
2	3	32	1	0.008166	0.5791
2	2	64	0	0.004287	0.6196
2	3	64	1	0.012879	1.0020
3	2	64	0	0.008423	1.0950
3	3	64	1	0.074510	12.4370
2	3	64	0	0.005203	0.5978
2	4	64	1	0.007040	1.2673
3	3	64	0	0.009528	0.8774

interval. As long as they have not converged to the new state, the wrong values are used in the calculation of the feedforward signal and this signal does not give the correct compensation. The number of parameters used in the feedforward model does not change the output variance very much.

6.3.3 Long term behaviour of the multivariable self-tuning controller

Long term regulation of a process for a constant set point is the most common type of control. A practical controller must be able to achieve good performance for an indefinite time under regulatory conditions, and at the same time give satisfactory control when disturbances or set point changes occur. Self-tuning regulators, based on

Aströms algorithm, have shown unsatisfactory long term behaviour, characterized by sudden bursts in the output, if the forgetting factor is lower than unity [18]. The inclusion of a forgetting factor is often necessary to prevent the parameter covariance matrix $\underline{\underline{P}}$ and the gain matrix $\underline{\underline{K}}$ from becoming very small, in which case it would be impossible to control disturbances or set point changes, especially for nonlinear systems. If the forgetting factor is less than one and the signal-to-noise ratio is large, which is typical for many chemical processes, the matrix $\underline{\underline{P}}$ and thus the gain matrix, will tend to increase with time as long as no disturbances occur. The parameter estimation algorithm is of the form:

new estimates = error term*gain + old estimates

Therefore, a large gain will give a disproportionately large change in the parameters for a small error. This change tends to build up, the error increases and the result is a burst in the output. Sometimes this change in the output will cause the algorithm to stabilize itself provided the $\underline{\underline{P}}$ matrix remains positive definite, because the large error will decrease both $\underline{\underline{P}}$ and $\underline{\underline{K}}$, and the output error will be reduced in the same way as a disturbance is controlled. Using a recursive square root identification method would ensure that $\underline{\underline{P}}$ is always positive definite and that good control is restored after such a disturbance. However, this

type of behaviour is always undesirable because it decreases the overall perfomance and because a change in the output away from the set point might make the product useless. The occurrence of a disturbance when the gain matrix is high will often trigger this unstable behaviour. Different solutions to this problem have been proposed. The forgetting factor could be made dependent on the output signal: equal to one if the output error is very small, and lower than one for larger output errors. Alternatively, the gain matrix could be kept artificially low by introducing a perturbation into the system from time to time. The second solution is not satisfactory because the plant is disturbed which causes a decrease in the overall control performance. Morris [18] proposed that the recursive least squares identification be replaced by a recursive learning algorithm after the output has stabilized. This effectively keeps the gain matrix low but the performance in the case of disturbances and set point changes is decreased, especially for highly nonlinear systems.

The bursting phenomenon is illustrated in Figures 14a, 14b and 14c for zero noise and a forgetting factor of 0.9, deliberately chosen smaller than it should be in order to trigger the abnormal behaviour without having to conduct the simulation for too long a time. Figure 14a shows that after 280 minutes the output error becomes very large and Figures 14b and 14c indicate that simultaneously the predicted parameters and the diagonal elements of the covariance

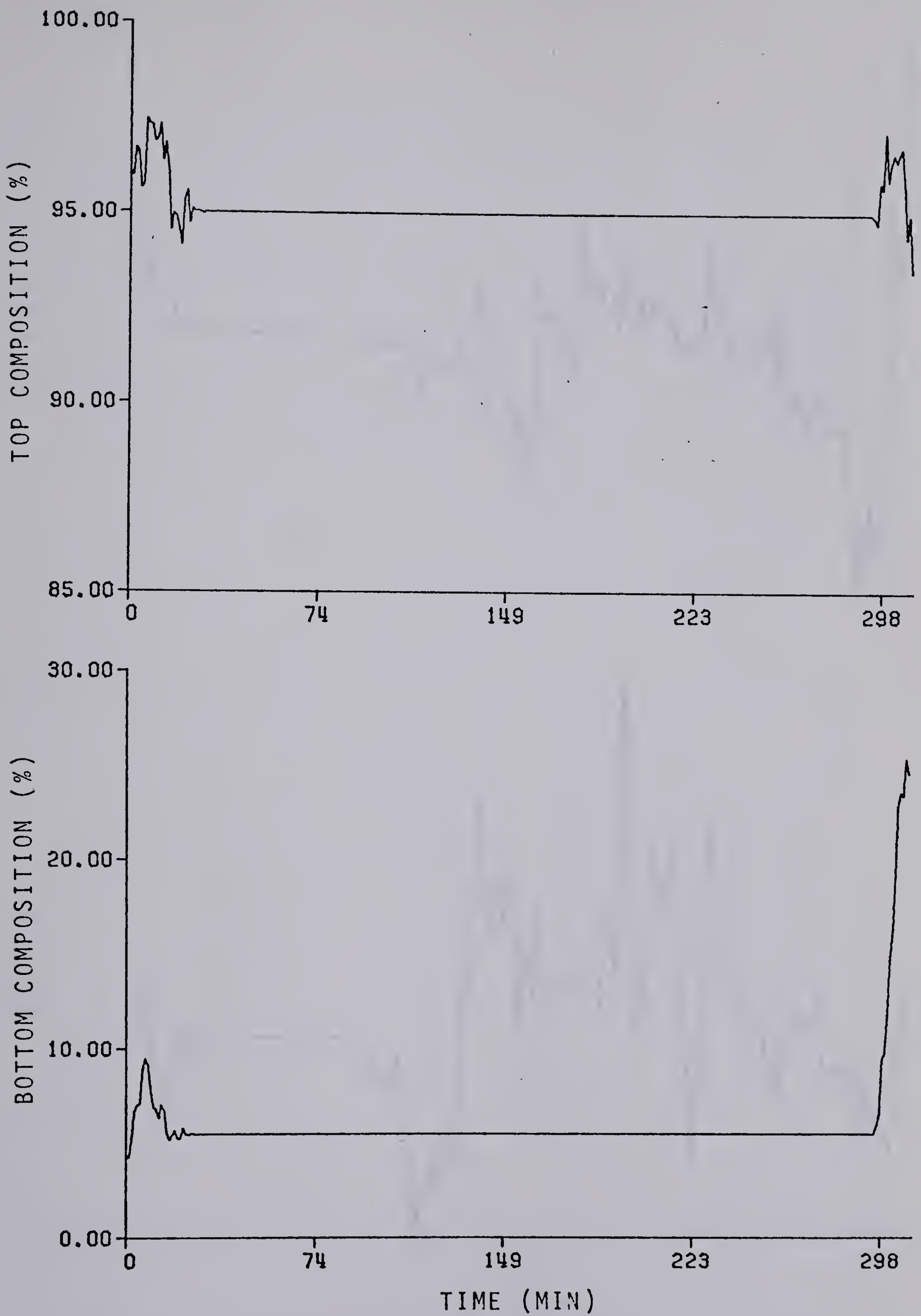


Figure 14a: Output behaviour for regulatory control using a forgetting factor of 0.9.

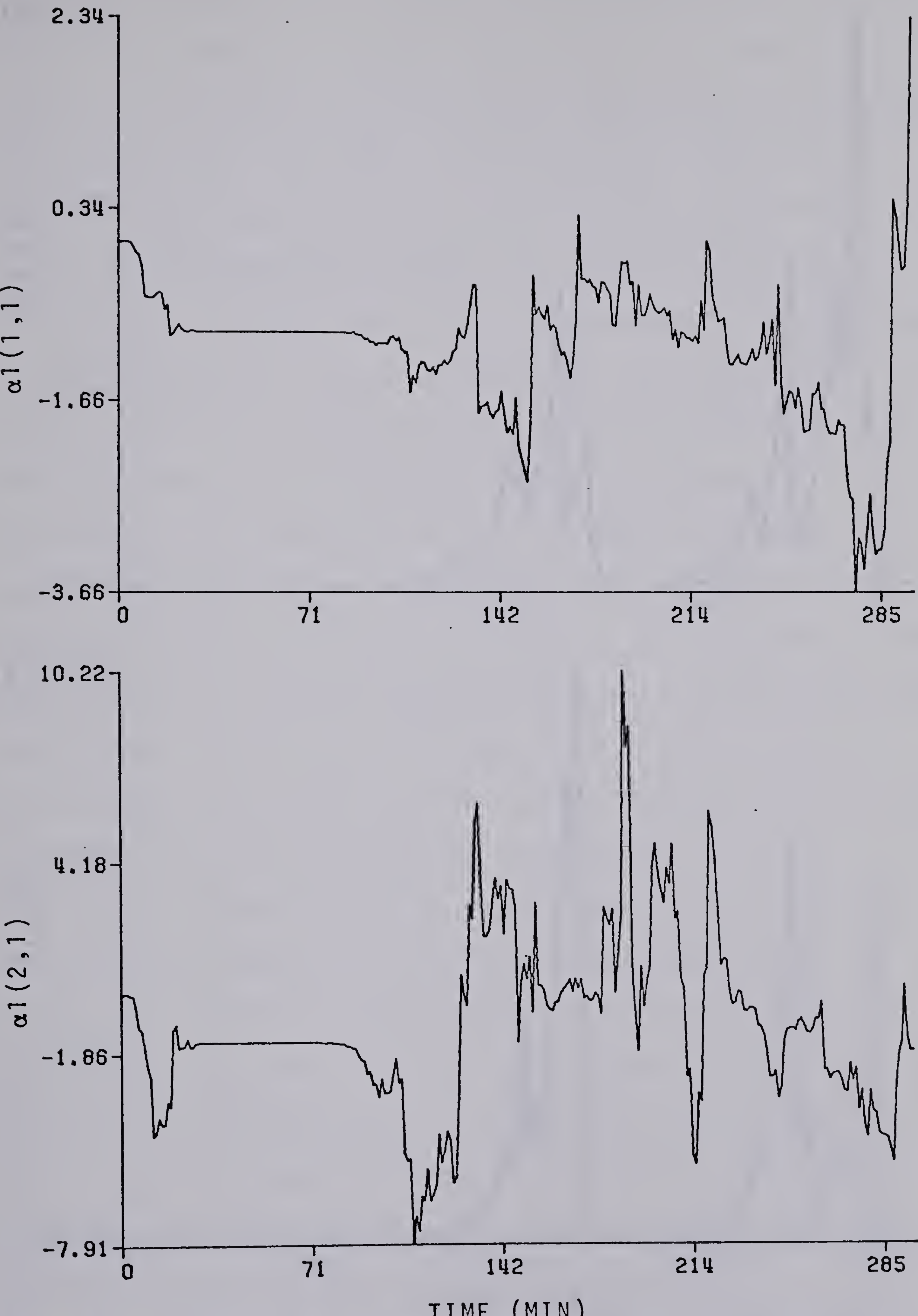


Figure 14b: Behaviour of α1(1,1) and α1(2,1) for regulatory control using a forgetting factor of 0.9.

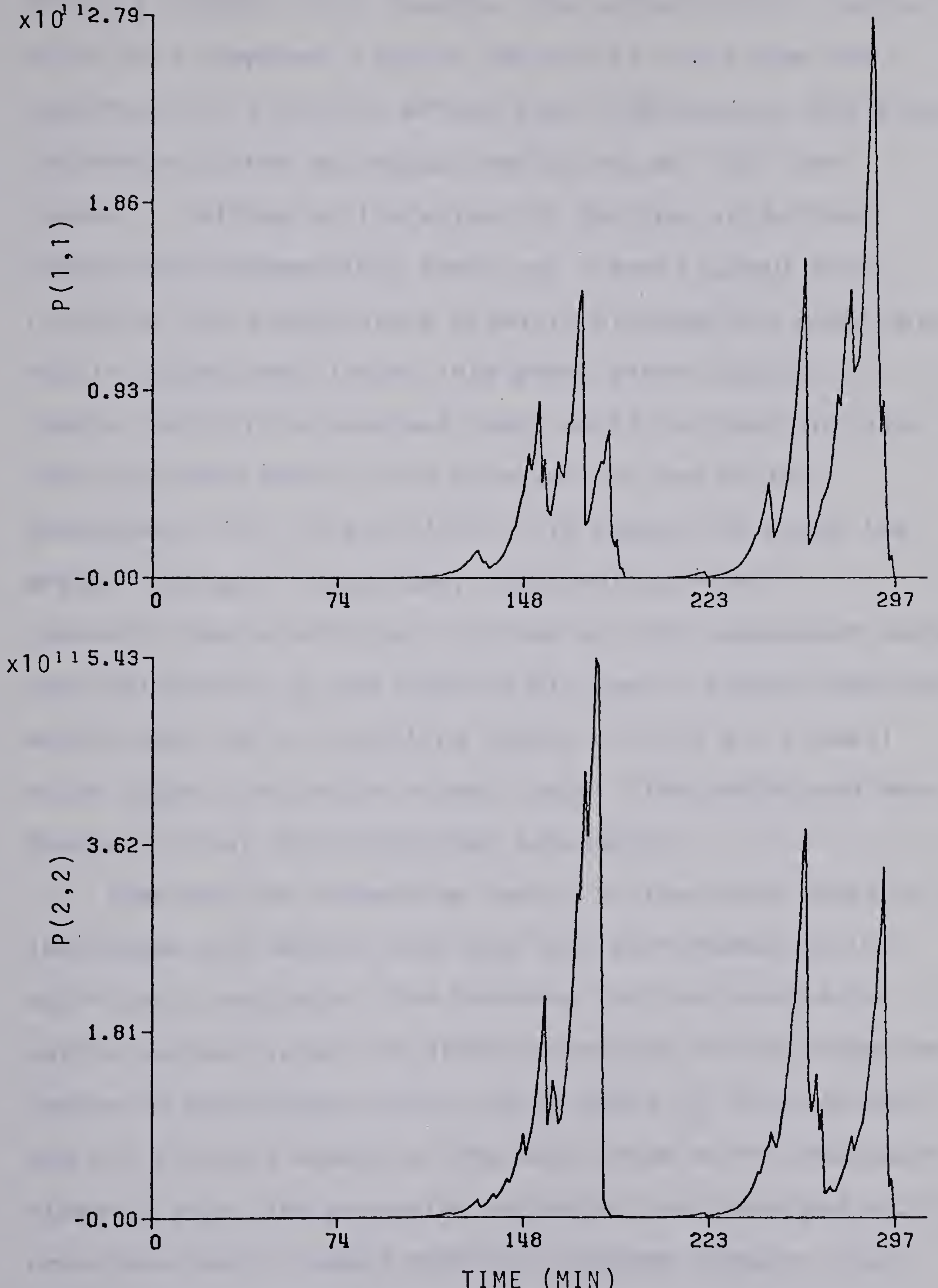
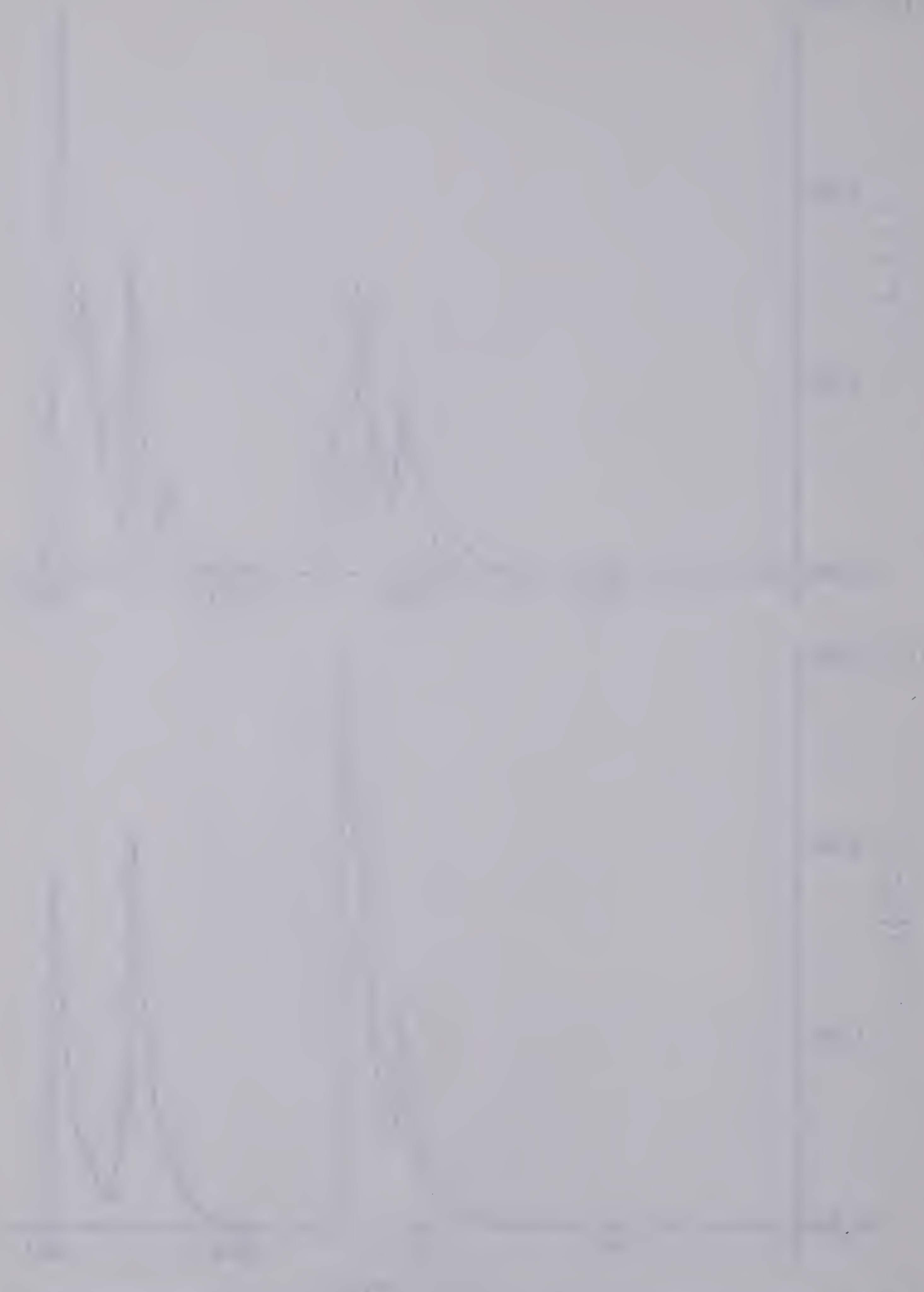


Figure 14c: Behaviour of P(1,1) and P(2,2) for regulatory control using a forgetting factor of 0.9. The steady state level is approximately 1000 for P(i,i).



matrix $\underline{\underline{P}}$ undergo large changes. The system did not recover after this happened. Figures 14b and 14c also show the occurrence of a similar effect after 185 minutes, but Figure 14a does not show any output variations at that time. However, listings of the values of the top and bottom compositions showed that there was a small output error (0.02% of the steady state values). Although the covariance matrix became very large, this small error could still reduce the gain to a normal level until the next increase in the covariance matrix. The behaviour of two of the parameters, $\alpha 1(1,1)$ and $\alpha 1(2,1)$, in Figure 14b shows the gradual change in magnitude, the bursting effect characterized by wild oscillations and the subsequent damped oscillations for a short period of time. A similar behaviour would occur for a forgetting factor of 0.99 and a small noise signal, but after a much longer time period and more gradually than in the previous simulation.

Adapting the forgetting factor to the noise level of the system can improve the long term performance of the self-tuning regulator. The behaviour of the covariance matrix was monitored for different values of the forgetting factor in combination with a noise level $\sigma\underline{\underline{I}}$. For σ=0.0005 and ω=1.0, the elements of the covariance matrix decreased steadily after the parameter estimates had converged and the covariance matrix would eventually become singular. For ω=0.99, the elements increased as long as no disturbance other than the noise entered the system and this leads to

the bursting effect shown in Figures 14a, 14b and 14c. Ideally, the elements of the covariance matrix should not increase or decrease under regulatory control, and this objective was achieved for ω=0.999 and σ=0.0005.

6.3.4 Set point control

When the set point of a process is changed, the controller should ideally bring the output to this new level in one sampling interval without overshoot or offset. In practice some overshoot is usually allowed and the adaptation can extend over several time periods. For the distillation column, and for multivariable systems in general, an additional requirement is that the changes in one loop should not disturb or modify the output behaviour of the other loop.

Output and input signals for set point changes in the composition of the top product are shown in Figures 15a and 15b. The different response to positive and negative changes demonstrates the nonlinearity of the system. Because of interaction, both reflux flow and steam flow rate have to be adapted in order to satisfy the new conditions. The bottom loop was only slightly disturbed when the changes occurred and the maximum error was -0.45% for a 2% increase in top composition set point. However, another simulation showed that this error became 1.36% for a 3% decrease in set point which indicates the nonlinear character of the interaction. The initial parameters were those shown in Figures 13c and

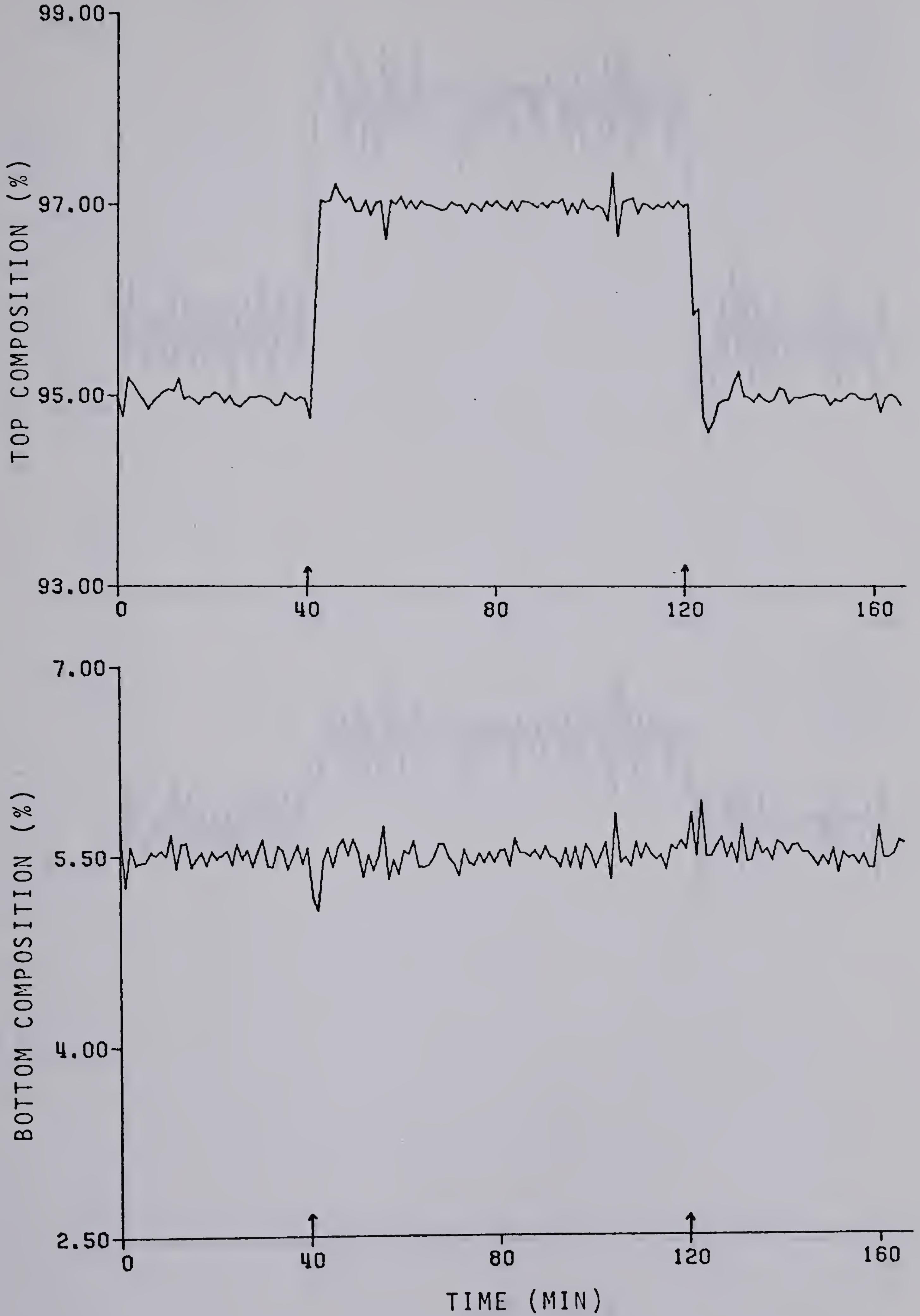


Figure 15a: Output control behaviour for +2% and -2% steps in the top composition set point.

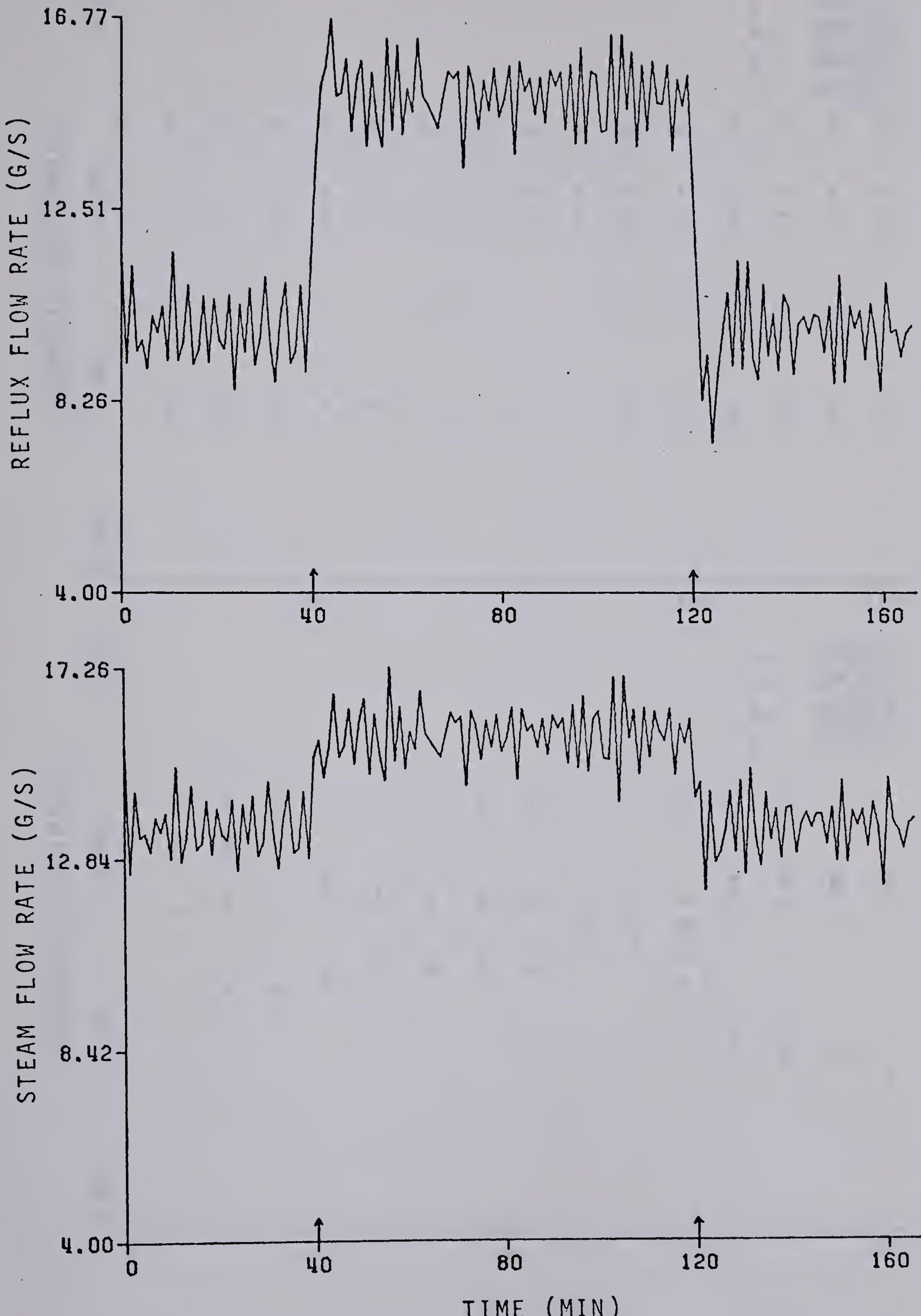
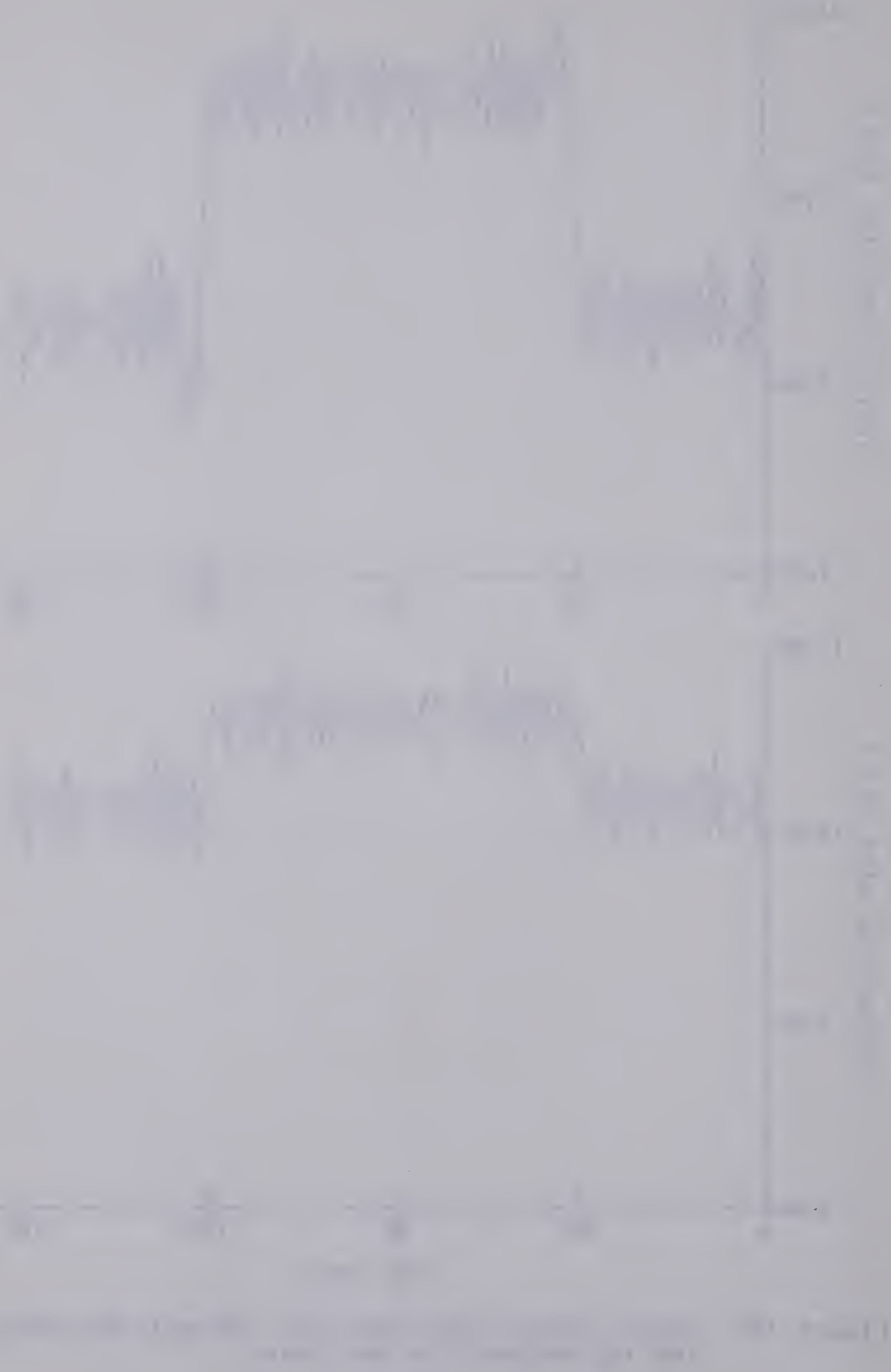


Figure 15b: Input control behaviour for +2% and -2% steps in the top composition set point.



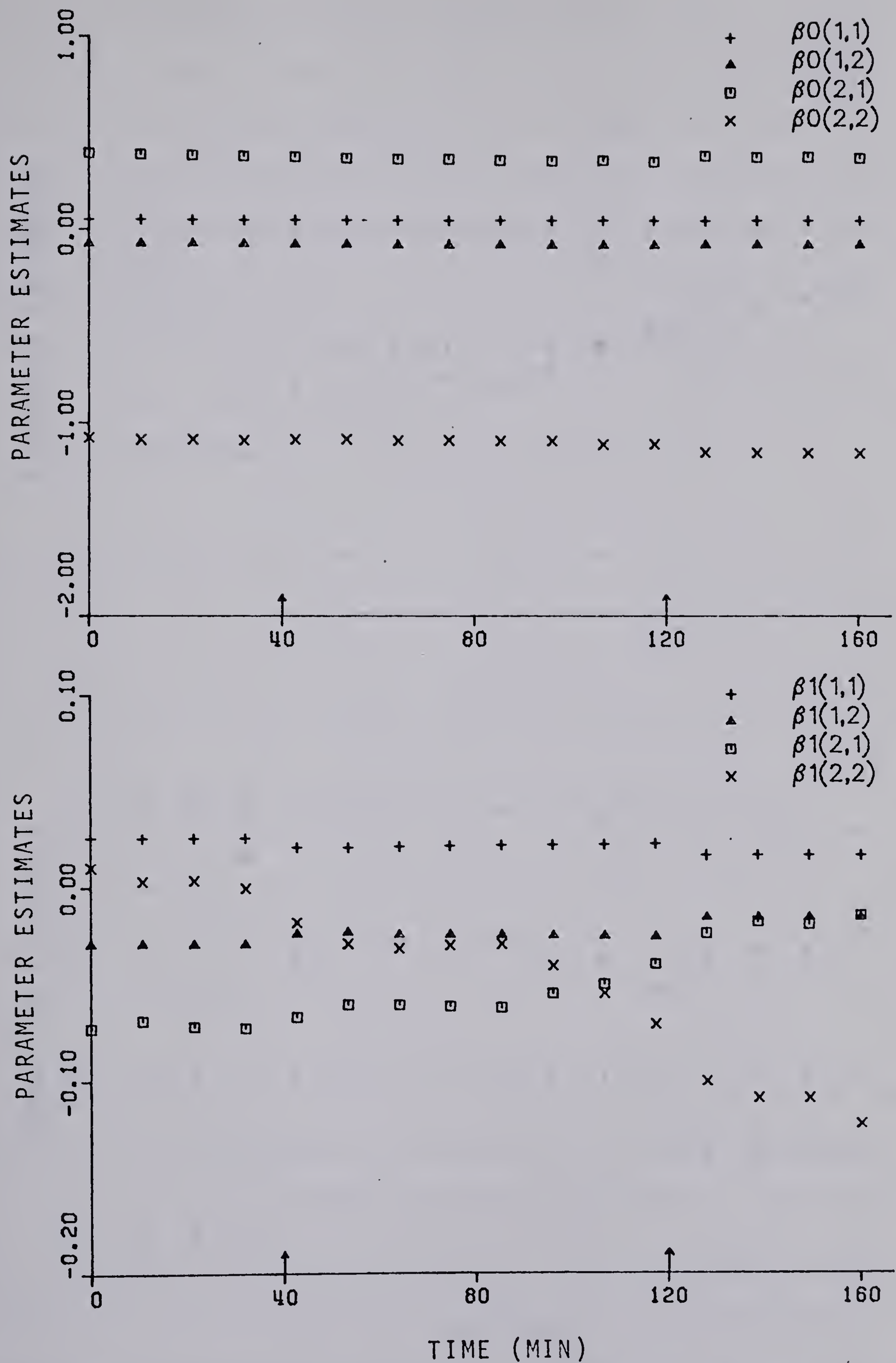


Figure 15c: Parameter adaption for +2% and -2% steps in the top composition set point.

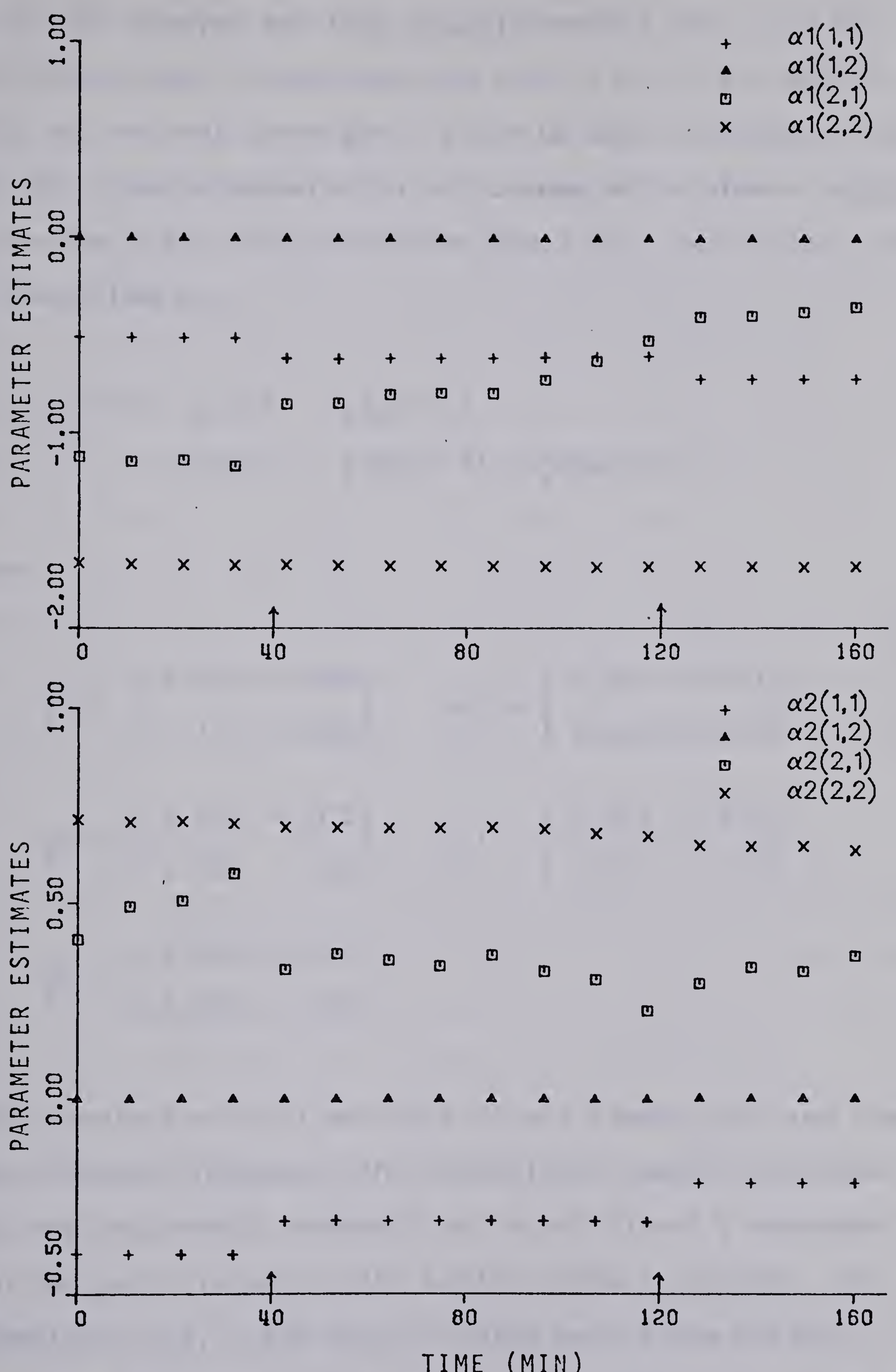


Figure 15d: Parameter adaption for +2% and -2% steps in the top composition set point.

13d at 165 minutes and they stayed constant until the set point change was introduced except for $\alpha1(2,2)$ and $\alpha2(2,2)$ which had not yet converged. As can be seen in Figures 15c and 15d, some parameters did not change while others adapted to the new state. The prediction model for the initial state was described by:

$$\underline{y}(t+1) = \underline{\underline{\alpha}}1\underline{y}(t) + \underline{\underline{\alpha}}2\underline{y}(t-1) + \underline{\underline{\beta}}0\Delta\underline{u}(t) + \underline{\underline{\beta}}1\Delta\underline{u}(t-1) + \underline{\underline{\beta}}2\Delta\underline{u}(t-2)$$

where

$$\underline{\underline{\alpha}}1 = \begin{vmatrix} -0.512 & -0.003 \\ -1.117 & -1.669 \end{vmatrix} \qquad \underline{\underline{\alpha}}2 = \begin{vmatrix} -0.397 & -0.001 \\ 0.434 & 0.711 \end{vmatrix}$$

$$\underline{\underline{\beta}}0 = \begin{vmatrix} 0.051 & -0.072 \\ 0.395 & -1.082 \end{vmatrix} \qquad \underline{\underline{\beta}}1 = \begin{vmatrix} 0.026 & -0.030 \\ -0.070 & 0.007 \end{vmatrix}$$

$$\underline{\underline{\beta}}2 = \begin{vmatrix} 0.008 & -0.010 \\ 0.055 & -0.196 \end{vmatrix}$$

The parameters $\alpha1(1,2)$ and $\alpha2(1,2)$ are almost zero and they never changed throughout the simulations, which indicates that the top product composition is not directly dependent upon the past history of the bottom product. However, the parameters $\alpha1(2,1)$ and $\alpha2(2,1)$ which relate the bottom product to previous values of the top product, are large and they fluctuate during set point changes thus indicating a

nonlinear relationship. The diagonal elements of the matrices, which indicate the dependence of an output upon its previous values, only vary when the corresponding set point is changed. The behaviour of the parameters relating inputs and outputs is different; the parameters of the first matrix $\underline{\underline{\beta}}0$ stay practically constant but those of $\underline{\underline{\beta}}1$ and $\underline{\underline{\beta}}2$ adapt to certain changes. The top product composition appears to be almost linearily related to reflux flow rate and steam flow rate but the bottom product reacts in a nonlinear way to changes in both those flows.

The process behaviour for the case of bottom composition set point changes is shown in Figures 16a, 16b, 16c and 16d. The disturbance to the top product composition is smaller than 0.5% if the change in the set point is +2%. The parameters relative to the top product composition stayed constant but the parameters in the second control loop changed when the set point was changed. Restricting the set point of the bottom product composition to a value lower than 4.5% decreased the controller performance in the top loop. For a set point lower than 4.5%, peaks with a magnitude larger than 0.5% appeared in the top product composition and these oscillations were not damped in time as can be seen in Figure 16a.

<u>6.3.5 Effect of the magnitude of a time delay on the control performance</u>

The effect of a time delay in the bottom loop on the

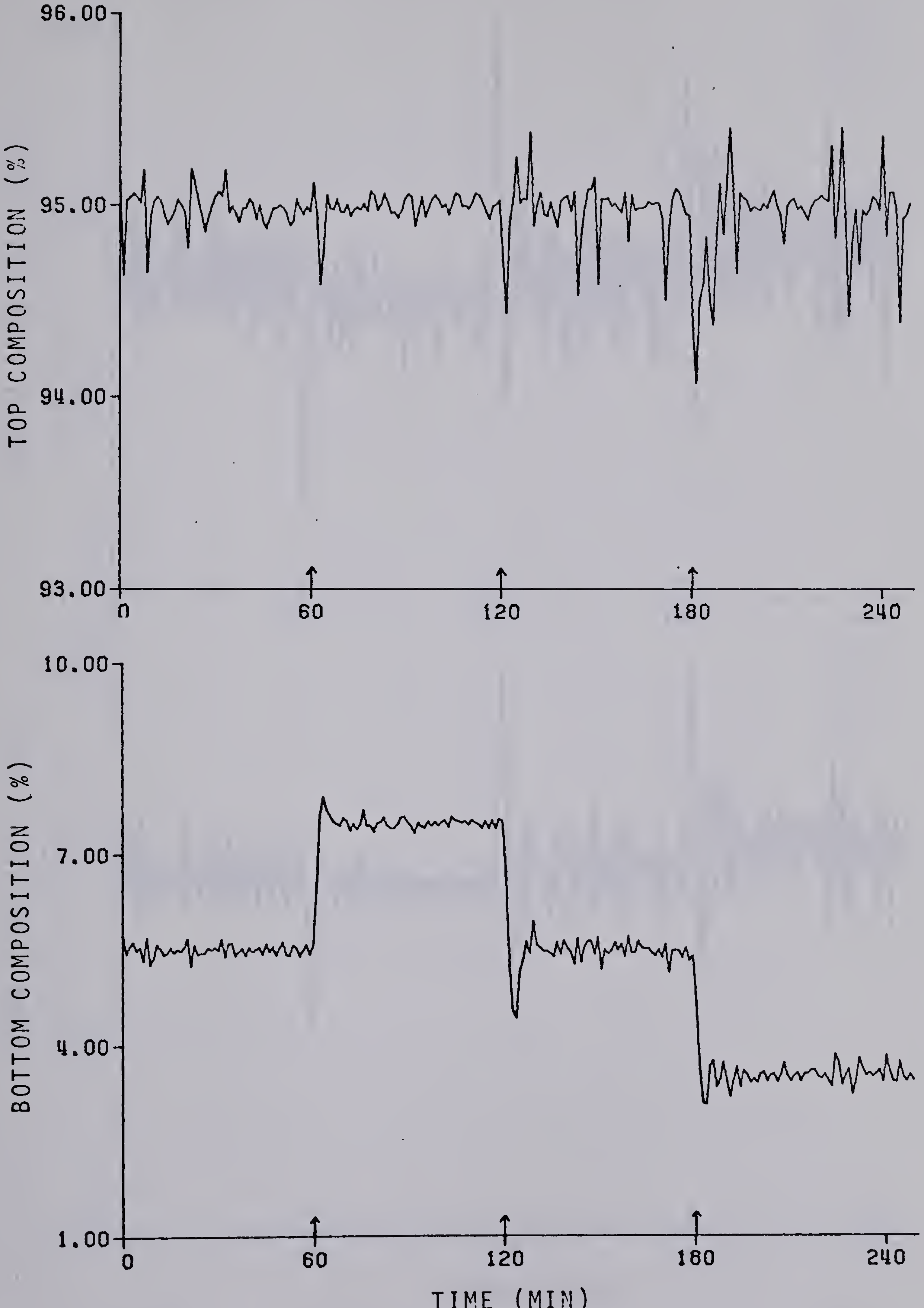


Figure 16a: Output control behaviour for set point changes in the bottom composition of magnitude +2%, -2% and -2% respectively.

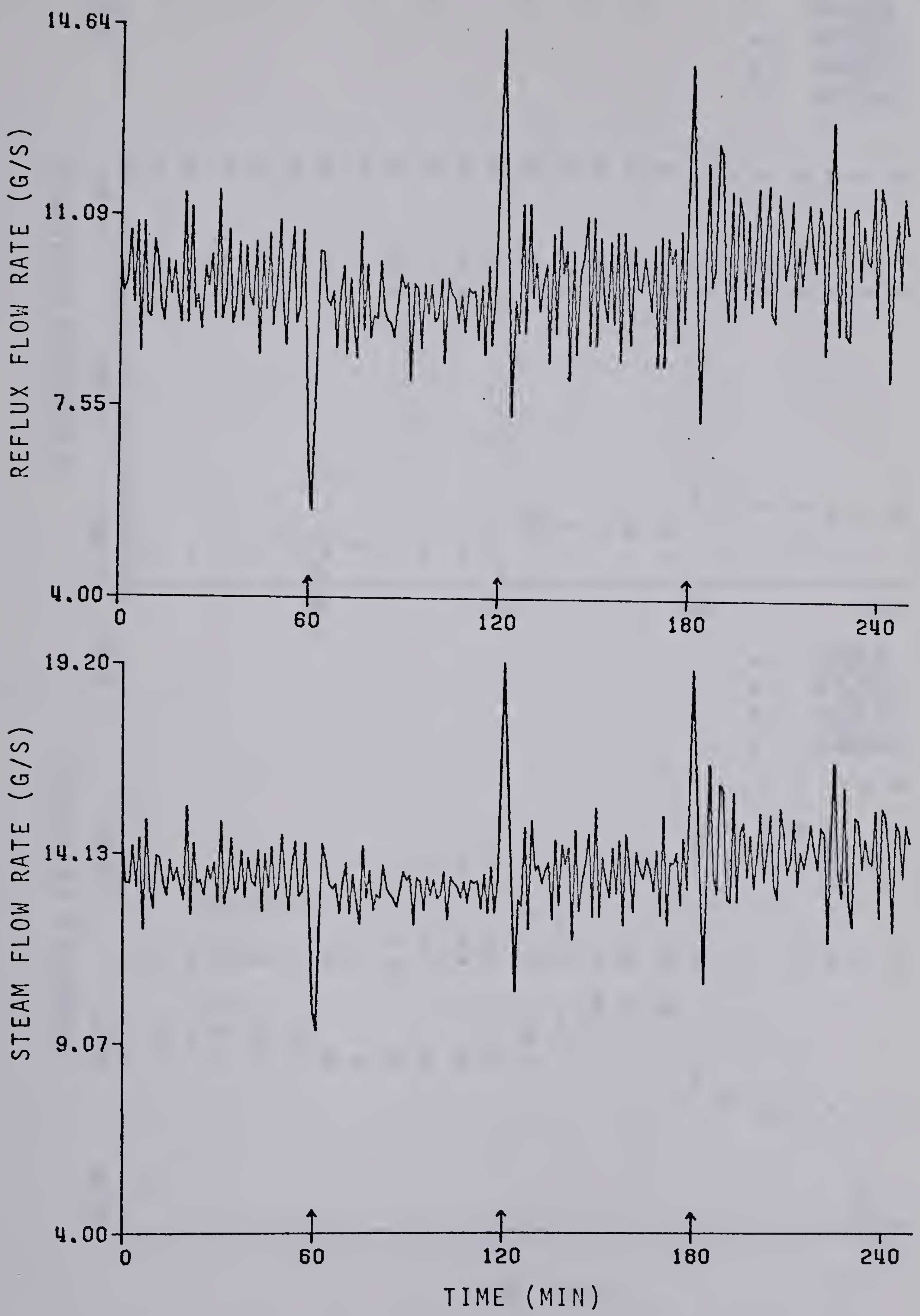


Figure 16b: Input control behaviour for set point changes in the bottom composition of magnitude +2%, -2% and -2% respectively.

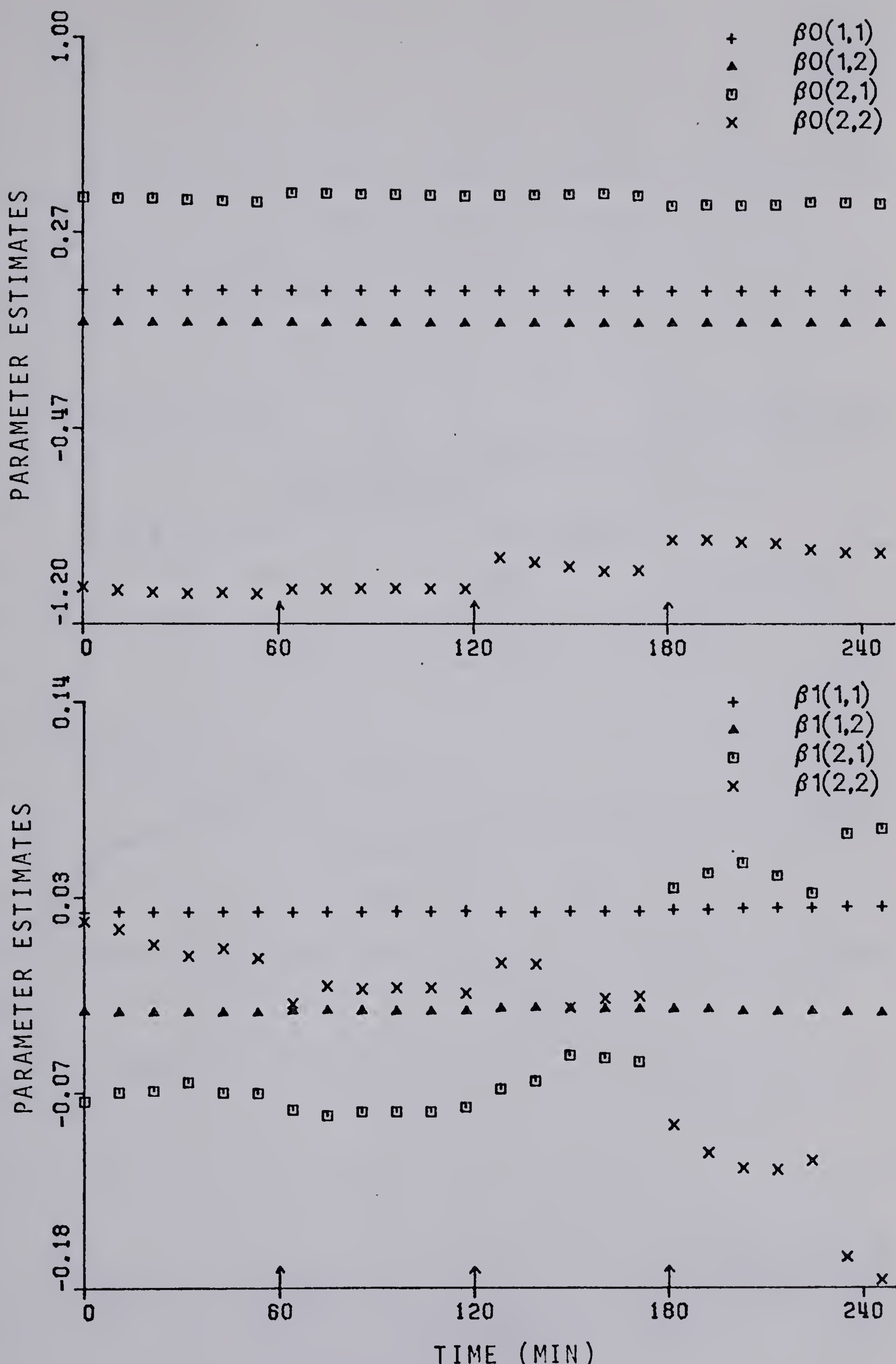


Figure 16c: Parameter adaption for set point changes in the bottom composition of magnitude +2%, -2% and -2% respectively.

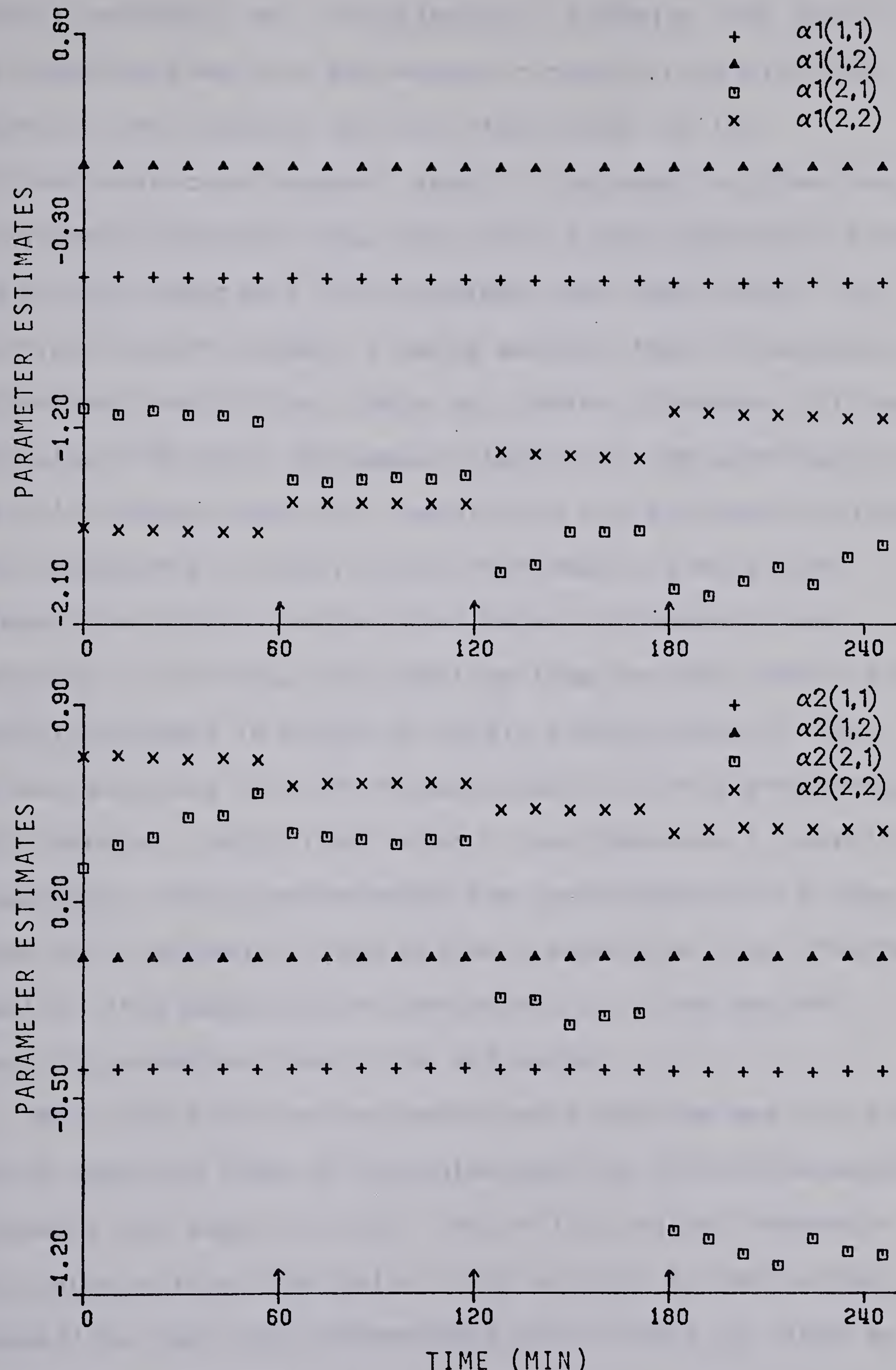


Figure 16d: Parameter adaption for set point changes in the bottom composition of magnitude +2%, -2% and -2% respectively.

control behaviour was investigated. In single loop control the sampling time must be chosen in conjunction with the dominant time constant and the time delay. In the multivariable case however, ideally the sampling time should be suitable for both loops and this is only possible if both loops have comparable time constants and time delays. For the distillation column, a delay smaller than 32 seconds in the bottom loop did not cause any control problems. Although the output variance increased slightly during start-up of the self-tuning algorithm, regulatory and set point control were comparable to the results obtained in simulations without time delay. As the time delay increased it was necessary to increase the sampling time and the order of the prediction model in order to obtain stable control. The maximum sampling time which gave stable control performance was 3 minutes. Using this value it was possible to obtain acceptable control performance for the column with a time delay of 80 seconds in the bottom composition loop. The best model in this case had the parameters $k=1$, $N=4$ and $M=4$. Thus, 32 parameters had to be estimated.

When the self-tuning controller algorithm was started with a sampling time of 3 minutes and the initial parameter estimates set equal to zero, the initial output behaviour of the column with a time delay of 80 seconds in the bottom composition loop was unacceptable with errors as large as 20% for both top and bottom composition. However, after approximately 60 minutes when the parameters had converged,

regulatory control of the column was comparable to the control performance obtained when using a smaller sampling interval and no time delay in the bottom loop.

Control in the case of a disturbance caused by an increase of 20% in the feed flow rate is shown in Figures 17a and 17b; the initial parameters were those obtained in the previous simulation after 60 minutes, and noise with a covariance matrix of $0.0005\underline{\underline{I}}$ was added to the output signals. Control of the bottom composition was good but the error in the top composition was 4.5% immediately after the disturbance and it took as long as 400 minutes to damp the subsequent oscillations which had a magnitude of 1%.

Control of the column for set point changes of 2% in the top composition is shown in Figures 18a and 18b. The maximum error in the bottom composition was 0.6%, as compared to 0.45% in Figure 15a where the sampling time was 64 seconds. The response of the top composition to the set point changes was also good with a maximum error of 0.8% as compared to 0.4% in Figure 15a. Set point changes of 1.5% in the bottom composition caused errors of 2.5% in the top composition but the bottom composition response was as good as when using a small sampling time and no time delay. The behaviour of output and input signals is shown in Figures 19a and 19b respectively.

It is clear from these simulations that self-tuning control of the column with a large time delay in the bottom loop and using a large sampling time is adequate for control

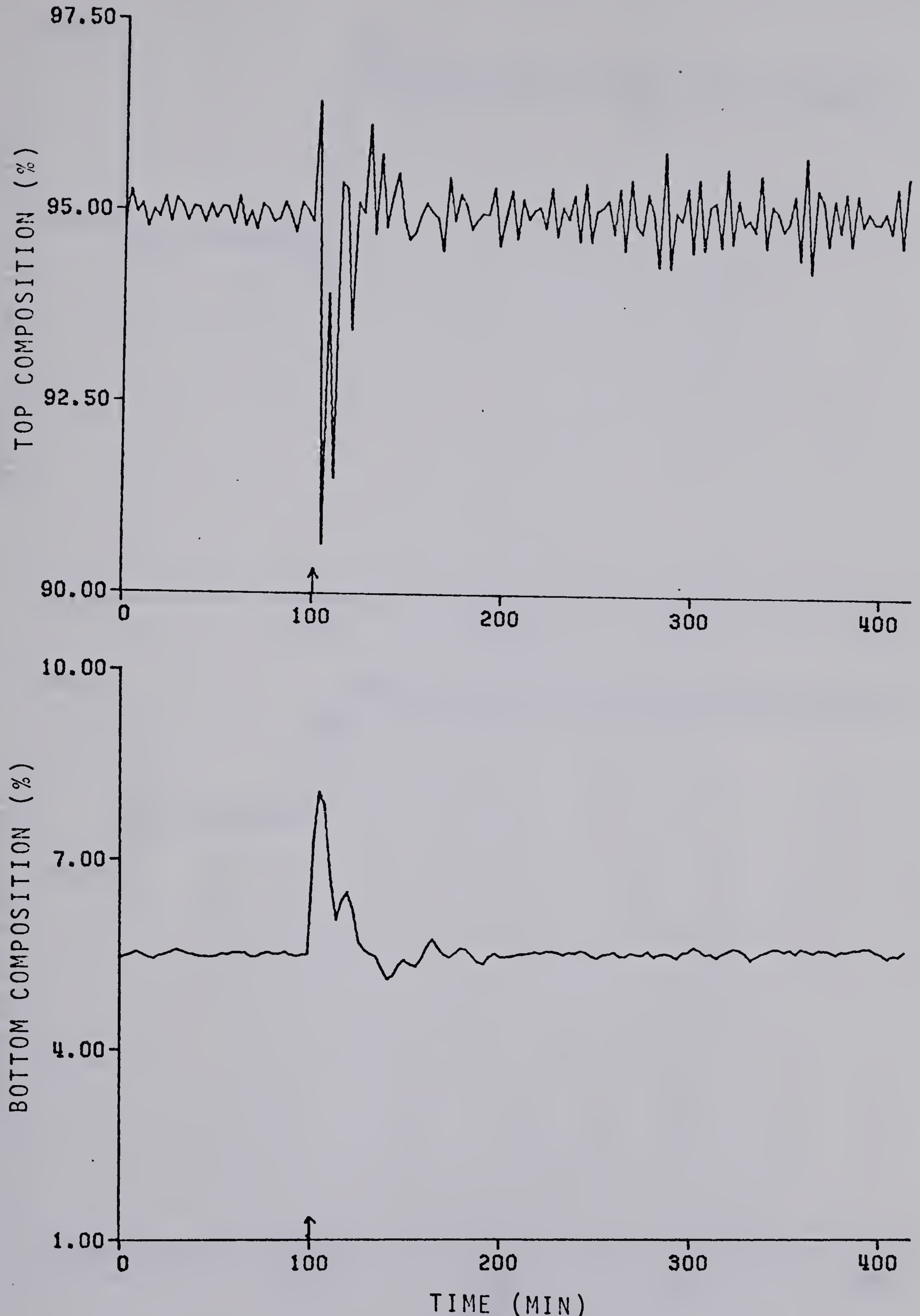


Figure 17a: Output control behaviour for a step increase of 20% in the feed flow rate using a time delay of 80 seconds in the bottom loop.

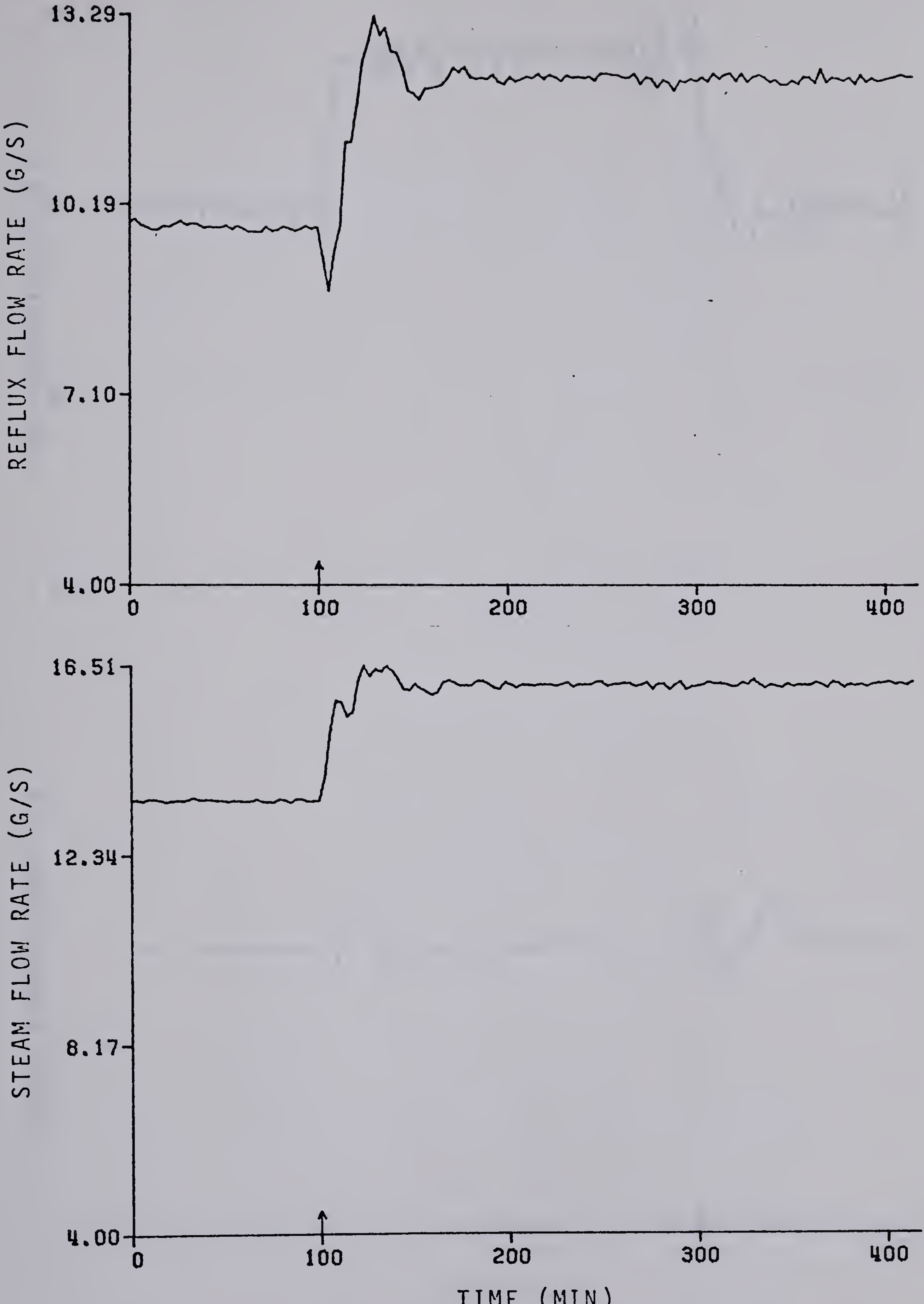
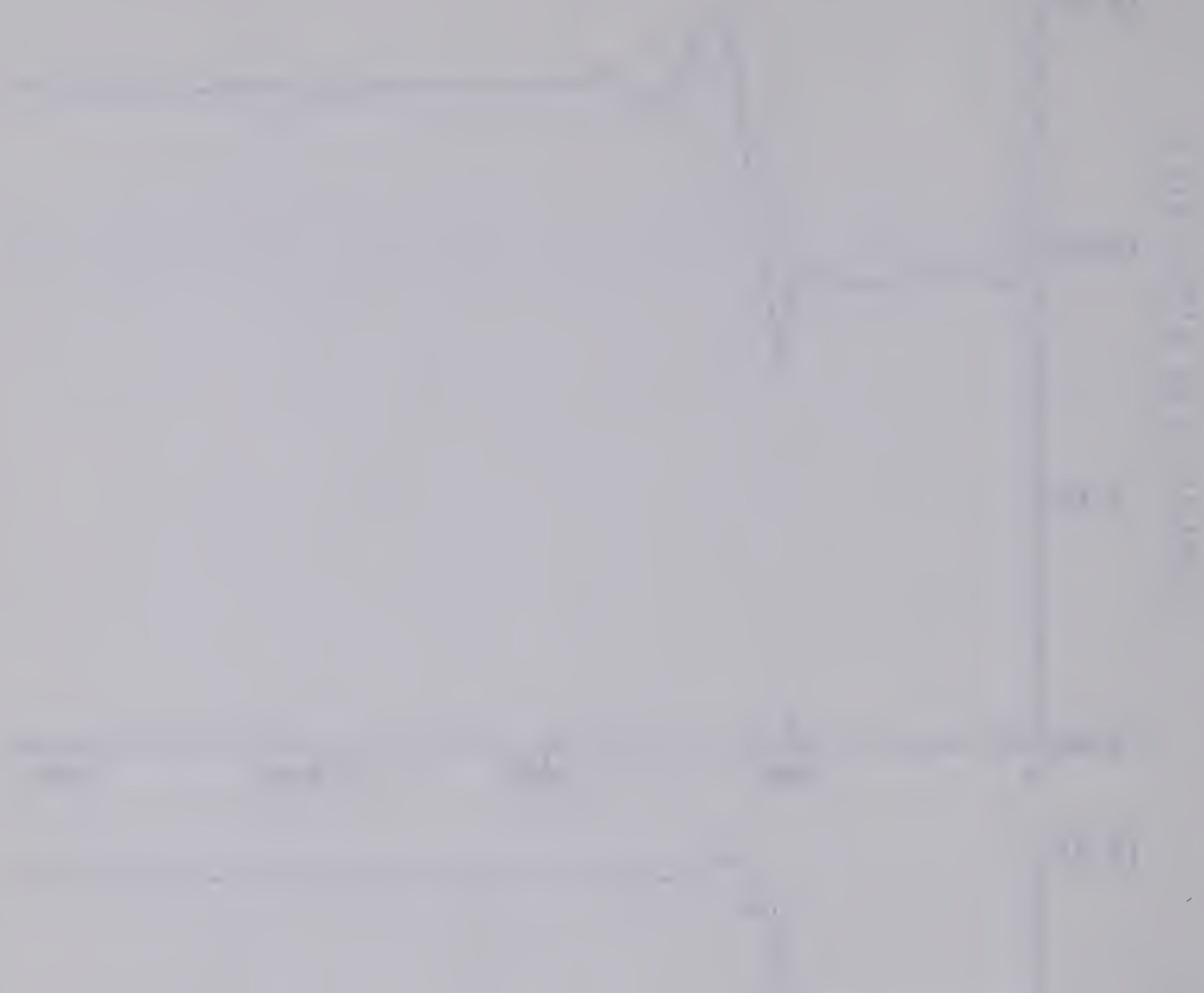


Figure 17b: Input control behaviour for a step increase of 20% in the feed flow rate using a time delay of 80 seconds in the bottom loop.



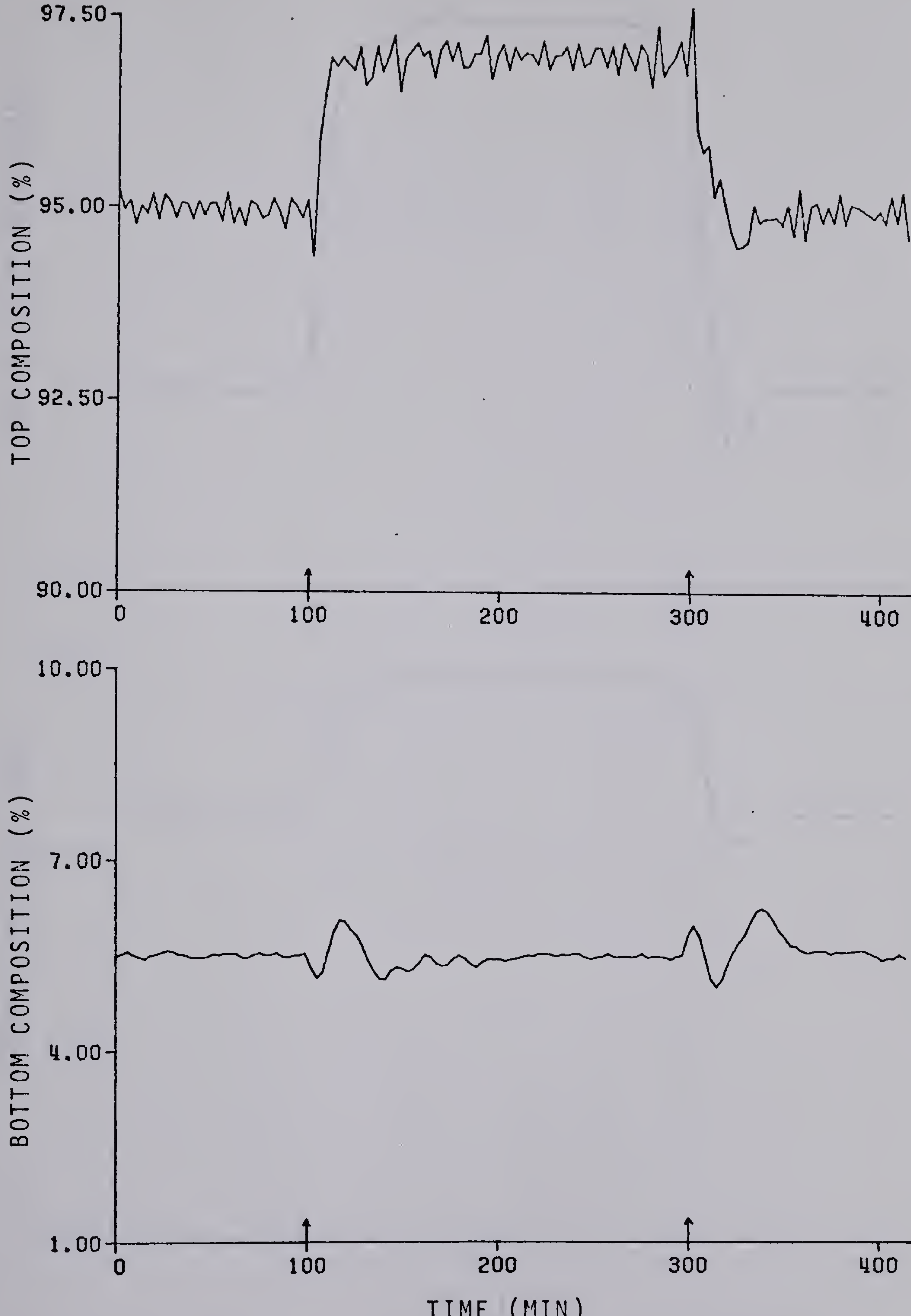
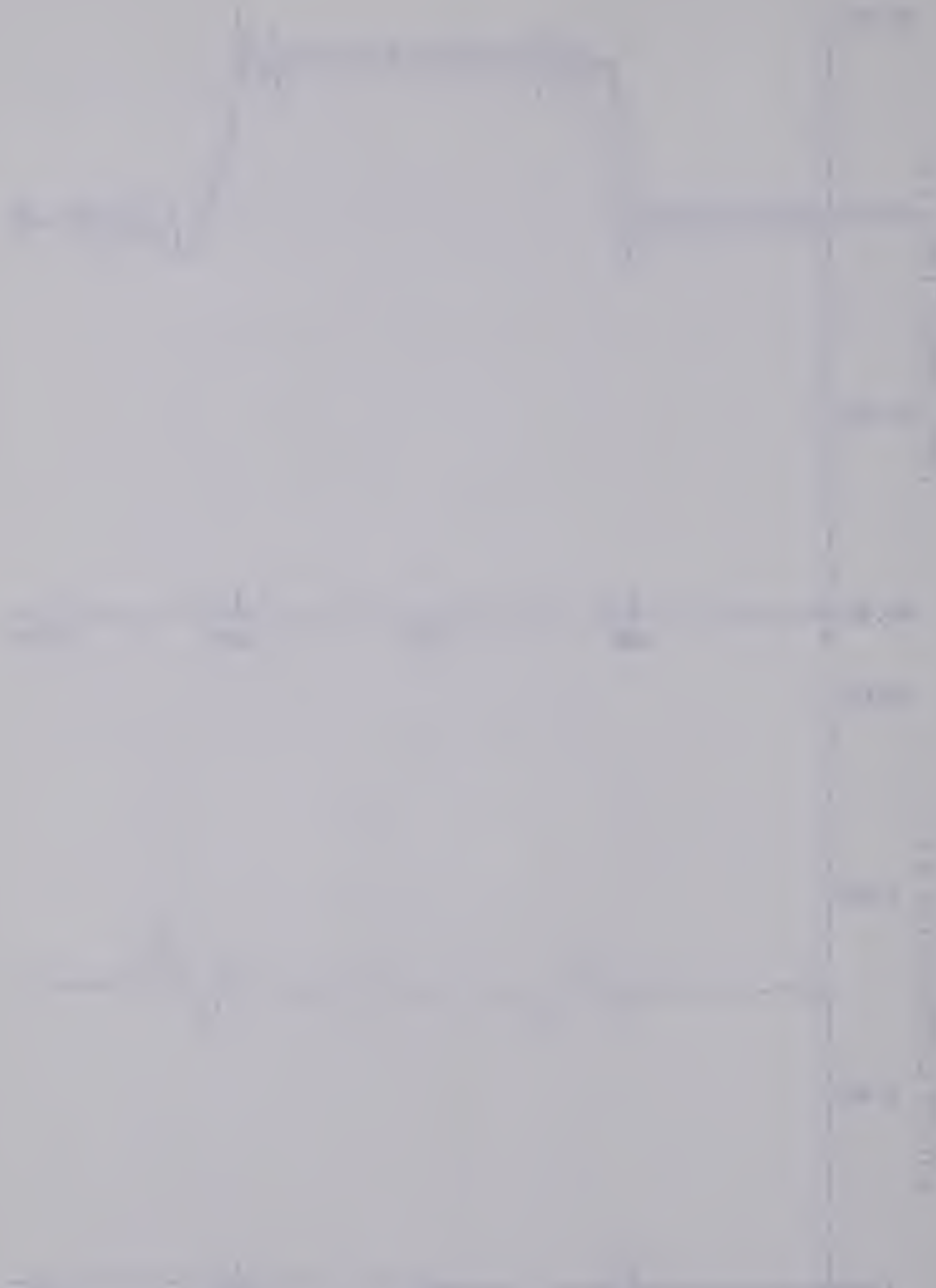


Figure 18a: Output control behaviour for step changes of 2% in the top composition set point using a time delay of 80 seconds in the bottom loop.



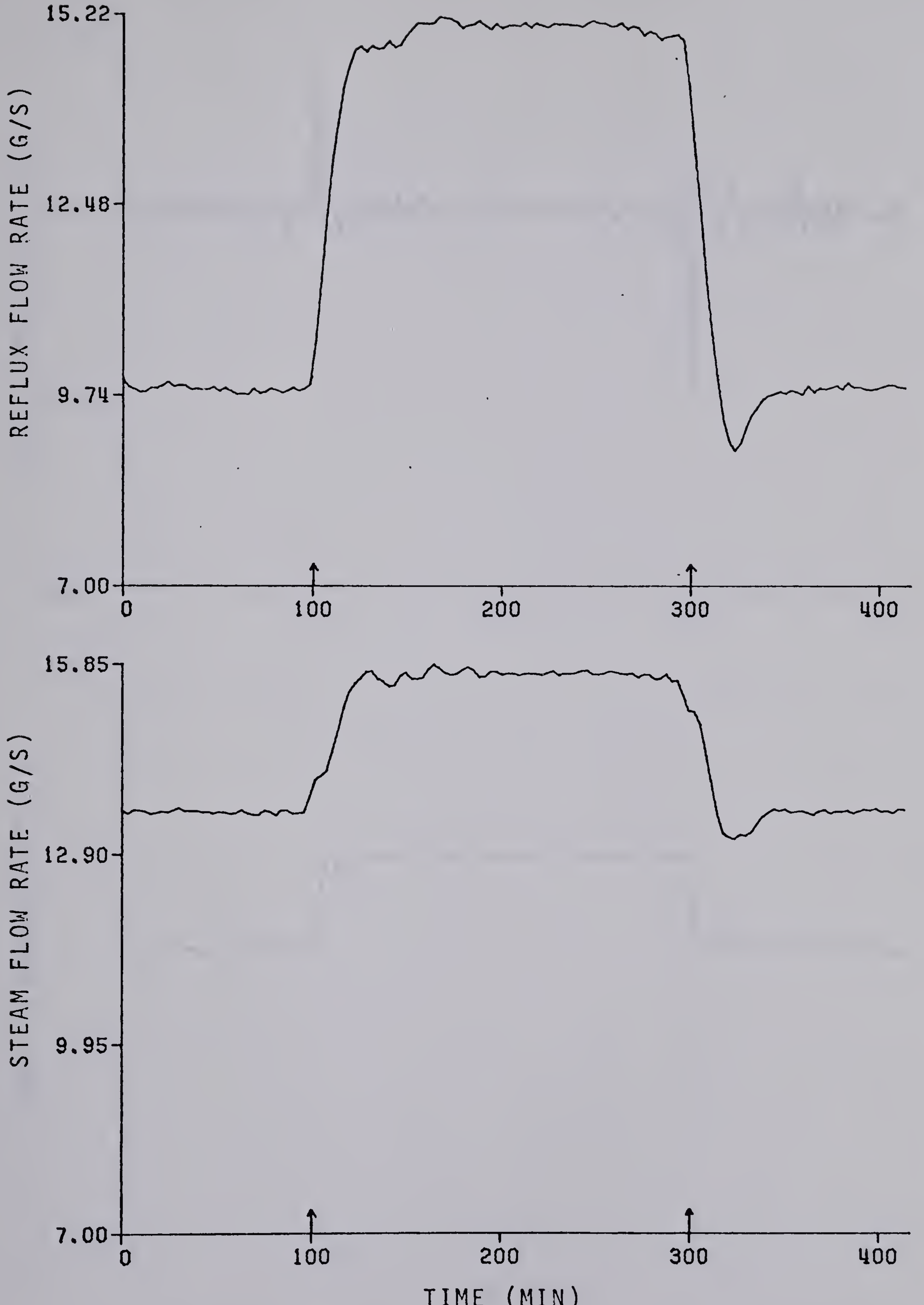


Figure 18b: Input control behaviour for step changes of 2% in the top composition set point using a time delay of 80 seconds in the bottom loop.

Figure 19a: Output control behaviour for step changes of 1.5% in the bottom composition set point using a time delay of 80 seconds in the bottom loop.

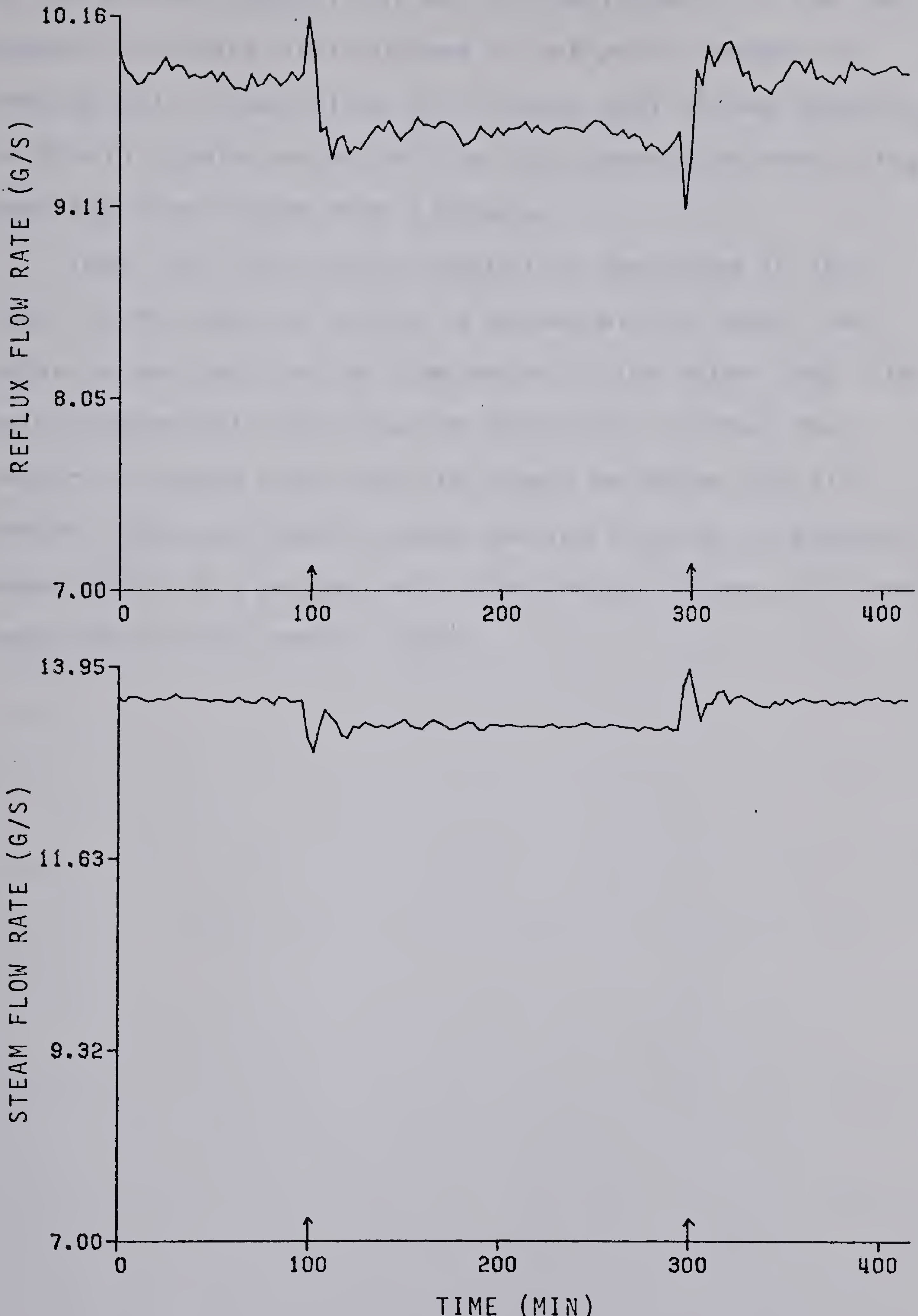


Figure 19b: Input control behaviour for step changes of 1.5% in the bottom composition set point using a time delay of 80 seconds in the bottom loop.

of the bottom composition, but the performance of the top composition after disturbances or set point changes is unacceptable. Simulations also showed that it was impossible to obtain stable control of the top composition when using a sampling time larger than 3 minutes.

Thus, the self-tuning controller described in this study is not able to control a column with a large time delay in one loop and no time delay in the other loop. The main problem with this type of controller is that the prediction model must have the same time delay for all control loops and such a model can not provide an adequate description of a process with time delays of very different magnitude in its control loops.

7. CONCLUSIONS AND RECOMMENDATIONS

Multivariable self-tuning controllers based on Aströms basic single loop regulator have some advantages but their practical application is linked with many problems which require further investigation before serious consideration can be given to their use as a standard control technique. One advantage of the controllers is their conceptual simplicity. The algorithm involves recursive least squares estimation and minimum variance control. The scheme is easy to program but the implementation requires the determination of a set of parameters which must be related to the actual process and this is only possible when a good knowledge of the process is available. The main characteristics of the controller are the built-in identification, which allows for on-line parameter estimation while the controller is in effect, and its ability to adapt to parameter changes, which is useful in the control of nonlinear and time-varying processes.

Long term performance of the algorithm remains a problem since unless some precaution is taken, the algorithm may cause uncontrolled changes in the output, which is unacceptable in most cases. If the signal-to-noise level of the system is fairly constant, the forgetting factor can be chosen such that the covariance matrix of the parameters does not decrease or increase during regulatory control. Other methods involve a variable forgetting factor or the

replacement of the least squares estimator by a recursive learning algorithm after the parameters have converged. None of these solutions are satisfactory for all applications because they might reduce the performance of the closed loop system under certain circumstances. However, they do prevent the bursts in the output from happening.

As with single loop controllers, it was found useful to implement integral control action in order to avoid offset. It was also shown that the use of a fixed parameter matrix $\underline{\underline{\beta}}0$ should be avoided. Although many authors accept it as a standard feature of the self-tuning regulator, it is not necessary to fix a parameter matrix, or one parameter in the single input - single output case. Such a decision could easily lead to failure of the controller if a very good estimate of the matrix or parameter is not available. This is especially true for multivariable systems since the relationship between the real value of the parameter matrix and the estimated fixed value is very restricted. Furthermore, for nonlinear processes fixed values would prevent those parameters from adapting to set point changes, which might decrease the closed loop performance.

Adding feedforward compensation to the algorithm will improve the output performance after disturbances only if the relation between the output and the measurable disturbance is linear. For highly nonlinear processes, the parameters in the disturbance model are only correct for a certain set point of the disturbance signal, and thus the

calculated compensation after a change will not be satisfactory until those parameters have adapted to the new state. Simulations with the binary distillation column model revealed that this procedure is not faster than the adaptation of the parameters in the input-output model after a disturbance so no gain in control performance was achieved by implementing feedforward compensation. This behaviour would not result from a control scheme with fixed parameters.

The main shortcoming of multivariable self-tuning controllers appears to be in the control of systems containing a large time delay in one loop. It was impossible to design the prediction model for the column in such a way that it would adequately describe the whole process. The choice of the sampling time seemed to be the critical factor. A large value would have been suitable for the bottom composition loop that contains the time delay but the output behaviour of the top composition loop deteriorated as the sampling time increased. It would be most worthwile if it were possible to establish a multivariable self-tuning control scheme which is not restricted to one sampling time for all the loops and thus would be able to adopt an optimal choice for each loop in accordance with their time constants and time delay.

In this work, only results of the simulation of the binary distillation column controlled by a multivariable self-tuning controller have been presented. The next step

would be to implement and test the multivariable self-tuning controller on the pilot scale binary distillation column in the Department of Chemical Engineering. However, from the simulation results some major problems can be predicted.

(a) The controller algorithm requires a lot of computer memory because a large number of parameters have to be estimated which involves calculations with large matrices. A reduction in the amount of memory required could be obtained by programming the algorithm in the form described by Borisson [7].

(b) It was not possible to obtain stable control in the simulation of the binary distillation column when the bottom loop was simulated with a time delay of 3 minutes, which is the actual time delay for the pilot scale column (this delay is largely due to the sample analyzer in the bottom loop). Further investigation into this problem is required. A self-tuning scheme which allows either different sampling times or different time delays for all the control loops would solve this problem. Another possibility which should be investigated is to use a fast sampling rate and to predict the bottom composition k steps ahead where k is the time delay in terms of the sampling interval.

The self-tuning algorithm tested in this work was based on minimum variance control and least squares estimation. A scheme which includes input signals and set points in the performance index [8] might improve the control performance of the column. Another interesting scheme that may be worthy of consideration is that based on the pole placement technique [22]. A controller designed on this basis might be able to deal effectively with the time delay in the bottom loop of the binary distillation column.

8. REFERENCES

1. Aström K. J.: Introduction to Stochastic Control Theory. Academic Press, (1970)

2. Aström K. J., Wittenmark B.: Problems of Identification and Control. Journal of Mathematical Analysis and Applications, 34, 90-113, (1971)

3. Aström K. J., Wittenmark B.: On Self-Tuning Regulators. Automatica, 9, 185-199, (1973)

4. Aström K. J., Wittenmark B.: Analysis of a Self-Tuning Regulator for Non-Minimum Phase Systems. IFAC Symposium on Stochastic Control, Budapest, (1974)

5. Aström K. J., Borisson U., Ljung L., Wittenmark B. : Theory and Applications of Self-Tuning Regulators. Automatica, 13, 457-476, (1977)

6. Borisson U.: Self-Tuning Regulators: Industrial Application and Multivariable Theory. Report 7513, Lund Institute of Technology, Division of Automatic Control (1975)

7. Borisson U.: Self-Tuning Regulators for a Class of Multivariable Systems. Automatica, 15, 209-215, (1979)

8. Clarke D. W., Gawthrop P. J.: Self-Tuning Controller. Proc. IEE, 122 (9), (1975)

9. Clarke D. W., Gawthrop P. J.: Self-Tuning Control. Proc. IEE, 126 (6), (1979)

10. H. W. Kan: M. Sc. Thesis, University of Alberta, Edmonton, Alberta, (1980)

11. Keviczky L., Hetthessy J., Hilgers M., Kolostori J. : Self-Tuning Adaptive Control of Cement Raw Material Blending. Automatica, 14, 525-532, (1978)

12. Koivo H. N.: A multivariable Self-Tuning Controller. Report 31, Tampere University of Technology, Department of Electrical Engineering, (1979)

13. Liesch D. W.: Decoupled Feedforward-Feedback Control of a Binary Bistillation Column. M. Sc. Thesis, University of Alberta, Edmonton, Alberta, (1974)

14. Lieuson H.: Evaluation of the Self-Tuning Controller on a Binary Distillation Column. M. Sc. Thesis, University of Alberta, Edmonton, Alberta, (1980)

15. Ljung L., Wittenmark B.: Analysis of a Class of Adaptive Regulators. IFAC Symposium on Stochastic Control, Budapest, (1974)

16. McGinnis R. G.: Multivariable Control of a Binary Distillation Column. M. Sc. Thesis, University of Alberta, Edmonton, Alberta, (1972)

17. Morris A. J., Fenton T. P., Nazer Y.: Application of Self-Tuning Regulators to the Control of Chemical Processes. IFAC Congress on Digital Computer Applications to Process Control, Holland, (1977)

18. Morris A. J., Nazer Y.: Self-Tuning Controllers for Single and Multivariable Systems. Part I: Theoretical evaluation of the control strategy. Part II: Evaluation of controller performance.
Submitted to Automatica, (1978)

19. Pacey W. C.: Control of a Binary Distillation Column; an Experimental Evaluation of Feedforward and combined Feedforward-Feedback Control Schemes. M. Sc. Thesis, University of Alberta, Edmonton, Alberta, (1973)

20. Sastry V. A., Seborg D. E., Wood R. K. : An Application of a Self-Tuning Regulator to a Binary Distillation Column. Proc. JACC, 346-354, (1976)

21. Svrcek W. Y.: Binary Distillation Column Dynamics. Ph. D. Thesis, University of Alberta, Edmonton, Alberta, (1967)

22. Wellstead P. E., Prager D., Zanker P. :Pole Assignment Self-Tuning Regulator. Proc IEE, 126 (8), (1979)

23. Wittenmark B.: A Self-Tuning Regulator. Report 7311, Lund Institute of Technology, Division of Automatic Control (1973)

9. APPENDIX A

A number of FORTRAN programs were written for the simulation of linear multivariable systems and multivariable self-tuning control schemes. They were executed on the AMDAHL 470V/7 computer system at the University of Alberta.

S.SYSTEM: This program simulates a linear process with a maximum of 5 output and input signals. The process parameters are taken from a data file, which also contains the parameters of a noise filter (if desired), the initial set points and the initial covariance matrix for the self-tuning controller algorithm. The program allows for introduction of two disturbance signals and two set point changes during the simulation. It also allows for calculation of variable set points if the control objective is to make the average output over a time period equal to the specifications.
The control routine is one of the following:
STR, STR2, DSTR, FSTR
Other subroutines called by the program are GAUSS (returns a noise signal) and MINV (Scientific Subroutine Package library routine which calculates the inverse of a matrix).

STR: Subroutine which calculates the basic multivariable self-tuning controller.

STR2: Subroutine wich calculates the multivariable self-tuning control algorithm and which includes estimation of the first $\underline{\underline{B}}$-matrix parameters.

DSTR: Subroutine which calculates the multivariable self-tuning controller with integral control action.

FSTR: Subroutine which calculates the multivariable self-tuning controller with feedforward compensation.

CONTRL: Subroutine which provides a link between the simulation program of the binary distillation column and the multivariable self-tuning controller.

MSTR: Subroutine which calculates a multivariable self-tuning controller with integral action and feedforward compensation capability for the binary distillation column. All parameter matrices are estimated and initial values must be provided through the data file; this allows for the use of estimates from previous runs

which improves the start-up behaviour.

GAUSS: Random noise generator (instead of using this routine which creates a noise vector with diagonal covariance matrix, a library routine can be used which returns a noise vector with predetermined covariance matrix).

```
C       ****************************************************
C
C          S.SYSTEM SIMULATES A LINEAR SYSTEM WITH MAXIMUM
C       DIMENSION 5 AND MAXIMUM 60 PARAMETERS. THE SYSTEM
C       PARAMETERS ARE READ FROM A DATA FILE AND THE PARAMETERS
C       FOR DIMENSIONALIZATION OF THE MODEL ARE READ
C       FROM THE TERMINAL (INTERACTIVELY).
C
C       ****************************************************
C
        DIMENSION A(5,20),B(5,25),C(5,20),Y(5),U(5),E(5),AIE(5)
        DIMENSION X(60),ZDUM(5,5),P(5,60),R(60,60),RAIE(5)
        DIMENSION YR(5),YA(5),PO(5,5),YRV(5,200),DU(5)
        DIMENSION DUDUM(5,10),YDUM(5,5),UDUM(5,10),EDUM(5,5)
        REAL LAMDA(5,5),G(5,20),SA(5),Z(5)
        INTEGER Q,F,FQ,D,IX(5),IZ(2),IYS(2)
C
C       INITIALIZATION
C
        DATA R,P,YRV,Y,U,DU,E/4920*0.0/,YDUM,UDUM,EDUM/100*0.0/
        DATA LAMDA,PO,Z,ZDUM,DUDUM/130*0.0/,X,YA/65*0.0/
        WRITE(6,1)
1       FORMAT('ORDER,TIME DELAY,PARAMETERS IN Y, U, E, Z')
        READ(5,2) Q,K,N,M,KE,F
2       FORMAT(6I6)
        WRITE(6,3)
3       FORMAT('READ A, B, C, G, YR, R(I,I), LAMDA(I,I)
       # FROM DATAFILE')
        NQ=N*Q
        MQ=M*Q+Q
        FQ=F*Q
        KEQ=KE*Q
        DO 400 I=1,Q
400     READ(7,4) (A(I,J),J=1,NQ)
        DO 700 I=1,Q
700     READ(7,4) (B(I,J),J=1,MQ)
        DO 600 I=1,Q
600     READ(7,4) (C(I,J),J=1,KEQ)
        DO 650 I=1,Q
650     READ(7,4) (G(I,J),J=1,FQ)
        READ(7,4) (YR(I),I=1,Q)
        LF=Q*(N+F+M+K)
        READ(7,4) (R(I,I),I=1,LF)
        READ(7,4) (LAMDA(I,I),I=1,Q)
4       FORMAT(10F12.5)
        WRITE(6,5)
5       FORMAT('CYCLE,IRAFT(=0,NO RAFT)')
        READ(5,2) NF,IRAFT
        WRITE(6,6)
6       FORMAT('TIME: DIST(1), DIST(2), SP(1), SP(2)')
        READ(5,2) IZ(1),IZ(2),IYS(1),IYS(2)
C
C       INITIALIZE BETA(0) MATRIX
```

```
      DO 100 I=1,Q
      DO 105 J=1,Q
      PO(I,J)=B(I,J)
105   CONTINUE
C
C     INITIALIZE VARIANCES AND NOISE PARAMETERS
C
      AIE(I)=0.0
      RAIE(I)=0.0
      IX(I)=333+17*I
      SA(I)=0.01
100   CONTINUE
      AM=0.0
      D=1+K
C
C     START SIMULATION
C
      DO 1000 ITEL1=1,32
C
C     CHECK FOR DISTURBANCE OR SET POINT CHANGE
C
      IF(ITEL1.GE.IZ(1)) Z(1)=1.0
      IF(ITEL1.GE.IZ(2)) Z(2)=1.0
      IF(ITEL1.GE.IYS(1)) YR(1)=1.0
      IF(ITEL1.GE.IYS(2)) YR(2)=1.0
      DO 500 NFC=1,NF
      DO 200 I=1,Q
      CALL GAUSS(IX(I),SA(I),AM,E(I))
200   CONTINUE
C
C     CALCULATE NEW OUTPUT SIGNALS
C
      DO 10 I=1,Q
      Y(I)=0.0
      DO 20 LY=1,N
      DO 20 J=1,Q
      Y(I)=Y(I)-A(I,J+(LY-1)*Q)*YDUM(J,LY)
20    CONTINUE
      MS=M+1
      DO 30 LU=1,MS
      DO 30 J=1,Q
      Y(I)=Y(I)+B(I,J+(LU-1)*Q)*UDUM(J,LU-1+D)
30    CONTINUE
      DO 35 LZ=1,F
      DO 35 J=1,Q
      Y(I)=Y(I)+G(I,J+Q*(LZ-1))*ZDUM(J,LZ)
35    CONTINUE
      DO 40 LE=1,KE
      DO 40 J=1,Q
      Y(I)=Y(I)+C(I,J+(LE-1)*Q)*EDUM(J,LE)
40    CONTINUE
      DO 50 J=1,Q
```

```
      Y(I)=Y(I)+LAMDA(I,J)*E(J)
50    CONTINUE
C
C     CALCULATE VARIANCES
C
      AIE(I)=AIE(I)+(Y(I)-YR(I))*(Y(I)-YR(I))
      YA(I)=YA(I)+Y(I)
      YRV(I,NFC+D)=0.0
C
C     IF RAFT, CALCULATE NEW SET POINT
C
      IF(IRAFT.EQ.0.OR.NFC.GE.NF+1-D) GOTO 12
      LM1=NFC+1
      LM2=NFC+D-1
      DO 60 J=LM1,LM2
      YRV(I,NFC+D)=YRV(I,NFC+D)+YRV(I,J)
60    CONTINUE
      YRV(I,NFC+D)=(-YRV(I,NFC+D)+YR(I)*NF-YA(I))/
     #(NF+1-NFC-D)
      GOTO 10
12    YRV(I,NFC+D)=YR(I)
10    CONTINUE
C
C     CALL CONTROLLER ALGORITHM
C
      CALL FSTR(N,M,F,D,Q,1.0,UDUM,YDUM,ZDUM,X,R,Y,Z,YRV,
     #U,P,NFC,PO)
C
C     UPDATE  MATRICES CONTAINING PAST SIGNALS
C
      DO 70 J=1,9
      JJ=10-J
      DO 70 I=1,Q
      UDUM(I,JJ+1)=UDUM(I,JJ)
      DUDUM(I,JJ+1)=DUDUM(I,JJ)
70    CONTINUE
      DO 80 J=1,4
      JJ=5-J
      DO 80 I=1,Q
      YDUM(I,JJ+1)=YDUM(I,JJ)
      EDUM(I,JJ+1)=EDUM(I,JJ)
      ZDUM(I,JJ+1)=ZDUM(I,JJ)
80    CONTINUE
      DO 90 I=1,Q
      U(I)=U(I)+DU(I)
      YDUM(I,1)=Y(I)
      UDUM(I,1)=U(I)
      EDUM(I,1)=E(I)
      ZDUM(I,1)=Z(I)
      DUDUM(I,1)=DU(I)
90    CONTINUE
C
```

```
C     WRITE VARIABLES AND PARAMETERS TO OUTPUT FILES
C
      WRITE(8,24) U(1),U(2),Z(1),R(1,1),R(4,4)
24    FORMAT(1H ,20(E12.5,1X))
      WRITE(9,24) Y(1),Y(2),AIE(1),AIE(2),RAIE(1),RAIE(2)
      DO 26 I=1,Q
      WRITE(10,25) (P(I,J),J=1,LF)
25    FORMAT(20F12.8)
26    CONTINUE
500   CONTINUE
C
C     RESET AVERAGE OUTPUT AFTER TIME PERIOD NF
C
      DO 1100 I=1,Q
      RAIE(I)=(YA(I)-YR(I)*NF)*(YA(I)-YR(I)*NF)/(NF*NF)
      YA(I)=0.0
      JJ=NF+D
      DO 1100 J=1,JJ
      YRV(I,J)=0.0
1100  CONTINUE
1000  CONTINUE
800   CONTINUE
      STOP
      END
```

```
C       *****************************************************
C
C          THE SUBROUTINE STR CALCULATES THE MULTIVARIABLE
C       SELF-TUNING CONTROL ALGORITHM. BETA(0) IS FIXED AND
C       THERE IS NO INTEGRAL OR FEEDFORWARD CONTROL ACTION.
C
C       *****************************************************
C
        SUBROUTINE STR(N,M,D,Q,W,UDUM,YDUM,X,R,Y,YRV,U,
       #P,NFC,PO)
        DIMENSION PO(5,5),ZZ(5),X(60),UDUM(5,10),YDUM(5,5),
       #HELP1(2),HELP2(2)
       #ALPHA(60),R(60,60)
        DIMENSION BETA(60,60),GAMMA(5),Y(5),U(5),DELTA(5,60),
       #PSI(5)
        DIMENSION EPS(5,60),P(5,60),YRV(5,200),PHI(2,2),
        INTEGER Q,D
C
C       INITIALIZE COUNTERS
C
        LF=Q*(N+M+D-1)
        LS=M+D-1
C
C       FORM VECTOR WITH INPUT AND OUTPUT SIGNALS
C
        DO 10 I=1,LS
        LI=Q*(I-1)
        DO 10 J=1,Q
        X(LI+J)=UDUM(J,D+I)
 10     CONTINUE
        LB=LS*Q
        DO 20 I=1,N
        LI=Q*(I-1)
        DO 20 J=1,Q
        X(LB+LI+J)=YDUM(J,D+I-1)
 20     CONTINUE
C
C       CALCULATE NEW PARAMETER COVARIANCE MATRIX
C
        DO 30 I=1,LF
        ALPHA(I)=0.0
        DO 30 J=1,LF
        ALPHA(I)=ALPHA(I)+R(I,J)*X(J)
 30     CONTINUE
        XO=0.0
        DO 40 I=1,LF
        XO=XO+X(I)*ALPHA(I)
        DO 40 J=1,LF
        BETA(I,J)=ALPHA(I)*ALPHA(J)
 40     CONTINUE
        XO=XO+W*W
        DO 50 I=1,LF
```

```
      DO 50 J=1,LF
      R(I,J)=(R(I,J)-BETA(I,J)/XO)/(W*W)
50    CONTINUE
C
C     CALCULATE PARAMETER UPDATES
C
      DO 60 I=1,Q
      GAMMA(I)=0.0
      ZZ(I)=0.0
      DO 70 J=1,LF
      GAMMA(I)=GAMMA(I)+P(I,J)*X(J)
70    CONTINUE
      DO 75 IL=1,Q
      ZZ(I)=ZZ(I)+PO(I,IL)*UDUM(IL,D)
75    CONTINUE
      GAMMA(I)=Y(I)-GAMMA(I)-ZZ(I)
      DO 60 J=1,LF
      DELTA(I,J)=GAMMA(I)*X(J)
      EPS(I,J)=0.0
60    CONTINUE
      DO 80 I=1,Q
      DO 80 J=1,LF
      DO 90 JJ=1,LF
      EPS(I,J)=EPS(I,J)+DELTA(I,JJ)*R(JJ,J)
90    CONTINUE
      P(I,J)=P(I,J)+EPS(I,J)
80    CONTINUE
C
C     FORM NEW VECTOR WITH PAST INPUT AND OUTPUT SIGNALS
C     FOR CALCULATING NEW INPUT SIGNALS
C
      DO 100 I=1,LS
      LI=Q*(I-1)
      DO 100 J=1,Q
      X(LI+J)=UDUM(J,I)
      X(LS*Q+J)=Y(J)
100   CONTINUE
      NN=N-1
      DO 110 I=1,NN
      LI=Q*I
      DO 110 J=1,Q
      X(LS*Q+LI+J)=YDUM(J,I)
110   CONTINUE
      DO 120 I=1,Q
      PSI(I)=0.0
      DO 130 J=1,LF
      PSI(I)=PSI(I)+P(I,J)*X(J)
130   CONTINUE
      PSI(I)=YRV(I,NFC+D)-PSI(I)
      DO 120 J=1,Q
      PHI(I,J)=PO(I,J)
120   CONTINUE
```

```
C
C      CALCULATE INVERSE OF BETA(0)
C
       CALL MINV(PHI,2,ZZ,HELP1,HELP2)
       DO 150 I=1,Q
       U(I)=0.0
       DO 140 J=1,Q
C
C      CALCULATE NEW INPUT SIGNALS
C
       U(I)=(U(I)+PHI(I,J)*PSI(J))
 140   CONTINUE
C
C     CONSTRAIN INPUT SIGNALS
C
 159   IF(U(I).GE.10.0) U(I)=10.0
       IF(U(I).LE.-10.0) U(I)=-10.0
 150   CONTINUE
       RETURN
       END
```

```
C      ********************************************************
C
C         THE SUBROUTINE STR2 CALCULATES A MULTIVARIABLE
C      SELF-TUNING CONTROL ALGORITHM AND ESTIMATES ALL
C      PARAMETERS.
C
C      ********************************************************
C
       SUBROUTINE STR2(N,M,D,Q,W,UDUM,YDUM,X,R,Y,YRV,U,P,
      #NFC,PO)
       DIMENSION PO(5,5),ZZ(5),X(60),UDUM(5,10),YDUM(5,5),
      #ALPHA(60),R(60,60),PR(5,5)
       DIMENSION BETA(60,60),GAMMA(5),Y(5),U(5),
      #DELTA(5,60),PSI(5)
       DIMENSION EPS(5,60),P(5,60),YRV(5,200),PHI(2,2),
      #HELP1(2),HELP2(2)
       INTEGER Q,D
C
C      INITIALIZE COUNTERS
C
       LF=Q*(N+M+D)
       LS=M+D
C
C      FORM VECTOR WITH PAST INPUT AND OUTPUT SIGNALS
C
       DO 10 I=1,LS
       LI=Q*(I-1)
       DO 10 J=1,Q
       X(LI+J)=UDUM(J,D+I-1)
 10    CONTINUE
       LB=LS*Q
       DO 20 I=1,N
       LI=Q*(I-1)
       DO 20 J=1,Q
       X(LB+LI+J)=YDUM(J,D+I-1)
 20    CONTINUE
C
C      CALCULATE NEW COVARIANCE MATRIX
C
       DO 30 I=1,LF
       ALPHA(I)=0.0
       DO 30 J=1,LF
       ALPHA(I)=ALPHA(I)+R(I,J)*X(J)
 30    CONTINUE
       XO=0.0
       DO 40 I=1,LF
       XO=XO+X(I)*ALPHA(I)
       DO 40 J=1,LF
       BETA(I,J)=ALPHA(I)*ALPHA(J)
 40    CONTINUE
       XO=XO+W*W
       DO 50 I=1,LF
       DO 50 J=1,LF
```

```
      R(I,J)=(R(I,J)-BETA(I,J)/XO)/(W*W)
50    CONTINUE
C
C     SAVE PREVIOUS VALUE OF CALCULATED BETA(0) MATRIX
C
      DO 60 I=1,Q
      DO 66 JR=1,Q
      PR(I,JR)=P(I,JR)
66    CONTINUE
C
C     CALCULATE UPDATED PARAMETER ESTIMATES
C
      GAMMA(I)=0.0
      DO 70 J=1,LF
      GAMMA(I)=GAMMA(I)+P(I,J)*X(J)
70    CONTINUE
      GAMMA(I)=Y(I)-GAMMA(I)
      DO 60 J=1,LF
      DELTA(I,J)=GAMMA(I)*X(J)
      EPS(I,J)=0.0
60    CONTINUE
      DO 80 I=1,Q
      DO 80 J=1,LF
      DO 90 JJ=1,LF
      EPS(I,J)=EPS(I,J)+DELTA(I,JJ)*R(JJ,J)
90    CONTINUE
      P(I,J)=P(I,J)+EPS(I,J)
80    CONTINUE
C
C     FORM NEW VECTOR WITH PREVIOUS INPUT AND OUTPUT
C     VALUES FOR USE IN INPUT CALCULATION
C
      LSS=M+D-1
      DO 100 I=1,LSS
      LI=Q*(I-1)
      DO 100 J=1,Q
      X(LI+J)=UDUM(J,I)
      X(LSS*Q+J)=Y(J)
100   CONTINUE
      NN=N-1
      DO 110 I=1,NN
      LI=Q*I
      DO 110 J=1,Q
      X(LSS*Q+LI+J)=YDUM(J,I)
110   CONTINUE
      LFF=Q*(N+M+D-1)
      DO 120 I=1,Q
      PSI(I)=0.0
      DO 130 J=1,LFF
      JJ=J+Q
      PSI(I)=PSI(I)+P(I,JJ)*X(J)
130   CONTINUE
```

```
      PSI(I)=YRV(I,NFC+D)-PSI(I)
      DO 120 J=1,Q
      PHI(I,J)=P(I,J)
120   CONTINUE
C
C     CALCULATE DETERMINANT OF NEW BETA(0) MATRIX
C     IF MATRIX SINGULAR, USE PREVIOUS VALUE
C     IF MATRIX NONSINGULAR, CALCULATE INVERSE
C
122   CALL MINV(PHI,2,ZD,HELP1,HELP2)
      IF (ZD.NE.0.0) GOTO 126
      DO 125 I=1,Q
      DO 125 J=1,Q
      PHI(I,J)=PR(I,J)
125   CONTINUE
      GOTO 122
C
C     CALCULATE NEW INPUT SIGNALS
C
126   DO 150 I=1,Q
      U(I)=0.0
      DO 140 J=1,Q
      U(I)=(U(I)+PHI(I,J)*PSI(J))
140   CONTINUE
C
C     CONSTRAIN INPUT SIGNALS
159   IF(U(I).GE.10.0) U(I)=10.0
      IF(U(I).LE.-10.0) U(I)=-10.0
150   CONTINUE
      RETURN
      END
```

```
C     **********************************************************
C
C        THE SUBROUTINE DSTR CALCULATES A MULTIVARIABLE SELF-
C     TUNING CONTROL ALGORITHM WITH INTEGRAL CONTROL ACTION.
C     THE MATRIX BETA(0) IS FIXED.
C
C     **********************************************************
C
      SUBROUTINE DSTR(N,M,D,Q,W,UDUM,YDUM,X,R,Y,YRV,U,P,
     #NFC,PO)
      DIMENSION PO(5,5),ZZ(5),X(60),UDUM(5,10),YDUM(5,5),
     #ALPHA(60),R(60,60)
      DIMENSION BETA(60,60),GAMMA(5),Y(5),U(5),DELTA(5,60),
     #PSI(5)
      DIMENSION EPS(5,60),P(5,60),YRV(5,200),PHI(2,2),
     #HELP1(2),HELP2(2)
      INTEGER Q,D
C
C     CALCULATE COUNTERS
C
      LF=Q*(N+M+D)
      LS=M+D-1
C
C     FORM VECTOR WITH PREVIOUS INPUTS AND OUTPUTS
C
      DO 10 I=1,LS
      LI=Q*(I-1)
      DO 10 J=1,Q
      X(LI+J)=UDUM(J,D+I)
10    CONTINUE
      LB=LS*Q
      NN=N+1
      DO 20 I=1,NN
      LI=Q*(I-1)
      DO 20 J=1,Q
      X(LB+LI+J)=YDUM(J,D+I-1)
20    CONTINUE
C
C     CALCULATE NEW COVARIANCE MATRIX
C
      DO 30 I=1,LF
      ALPHA(I)=0.0
      DO 30 J=1,LF
      ALPHA(I)=ALPHA(I)+R(I,J)*X(J)
30    CONTINUE
      XO=0.0
      DO 40 I=1,LF
      XO=XO+X(I)*ALPHA(I)
      DO 40 J=1,LF
      BETA(I,J)=ALPHA(I)*ALPHA(J)
40    CONTINUE
      XO=XO+W*W
      DO 50 I=1,LF
```

```
      DO 50 J=1,LF
      R(I,J)=(R(I,J)-BETA(I,J)/XO)/(W*W)
50    CONTINUE
C
C     CALCULATE NEW PARAMETER ESTIMATES
C
      DO 60 I=1,Q
      GAMMA(I)=0.0
      ZZ(I)=0.0
      DO 70 J=1,LF
      GAMMA(I)=GAMMA(I)+P(I,J)*X(J)
70    CONTINUE
      DO 75 IL=1,Q
      ZZ(I)=ZZ(I)+PO(I,IL)*UDUM(IL,D)
75    CONTINUE
      GAMMA(I)=Y(I)-GAMMA(I)-ZZ(I)
      DO 60 J=1,LF
      DELTA(I,J)=GAMMA(I)*X(J)
      EPS(I,J)=0.0
60    CONTINUE
      DO 80 I=1,Q
      DO 80 J=1,LF
      DO 90 JJ=1,LF
      EPS(I,J)=EPS(I,J)+DELTA(I,JJ)*R(JJ,J)
90    CONTINUE
      P(I,J)=P(I,J)+EPS(I,J)
80    CONTINUE
C
C     FORM VECTOR WITH INPUT SIGNALS AND OUTPUT SIGNALS FOR
C     COMPUTING OF THE CHANGES IN THE INPUT SIGNALS
C
      DO 100 I=1,LS
      LI=Q*(I-1)
      DO 100 J=1,Q
      X(LI+J)=UDUM(J,I)
      X(LS*Q+J)=Y(J)
100   CONTINUE
      DO 110 I=1,N
      LI=Q*I
      DO 110 J=1,Q
      X(LS*Q+LI+J)=YDUM(J,I)
110   CONTINUE
      DO 120 I=1,Q
      PSI(I)=0.0
      DO 130 J=1,LF
      PSI(I)=PSI(I)+P(I,J)*X(J)
130   CONTINUE
      PSI(I)=YRV(I,NFC+D)-PSI(I)
      DO 120 J=1,Q
      PHI(I,J)=PO(I,J)
120   CONTINUE
C
```

```
C     CALCULATE INVERSE OF BETA(0)
C
      CALL MINV(PHI,2,ZZ,HELP1,HELP2)
C
C     CALCULATE CHANGES IN CONTROL SIGNALS
C
      DO 150 I=1,Q
      U(I)=0.0
      DO 140 J=1,Q
      U(I)=U(I)+PHI(I,J)*PSI(J)
140   CONTINUE
C
C     LIMIT CHANGES IN CONTROL SIGNALS
C
      IF(U(I).GE.1.0) U(I)=1.0
      IF(U(I).LE.-1.0) U(I)=-1.0
150   CONTINUE
      RETURN
      END
```

```
C     ******************************************************
C
C        THE SUBROUTINE FRAFT CALCULATES THE MULTIVARIABLE
C     SELF-TUNING CONTROL ALGORITHM WITH FEEDFORWARD
C     COMPENSATION.
C     THE ROUTINE RECEIVES ALL PARAMETERS FROM THE CALLING
C     ROUTINE AND CAN BE CALLED BY S.SYSTEM.
C
C     ******************************************************
C
      SUBROUTINE FSTR(N,M,F,D,Q,W,UDUM,YDUM,ZDUM,X,R,Y,Z,
     #YRV,U,P,NFC,PO)
      DIMENSION PO(5,5),Z(5),X(60),UDUM(5,10),YDUM(5,5),
     #ALPHA(60),R(60,60)
      DIMENSION BETA(60,60),ZDUM(5,5),GAMMA(5),Y(5),U(5),
     #DELTA(5,60),PSI(5),ZZ(5)
      DIMENSION EPS(5,60),P(5,60),YRV(5,200),PHI(2,2),
     #HELP1(2),HELP2(2)
      INTEGER Q,D,F,FF
C
C     INITIALIZE COUNTERS
C
      LF=Q*(N+F+M+D-1)
      LS=M+D-1
      DO 10 I=1,LS
      LI=Q*(I-1)
C
C     FORM VECTOR WITH PROCESS SIGNALS
C
      DO 10 J=1,Q
      X(LI+J)=UDUM(J,D+I)
10    CONTINUE
      LB=LS*Q
      DO 20 I=1,N
      LI=Q*(I-1)
      DO 20 J=1,Q
      X(LB+LI+J)=YDUM(J,D+I-1)
20    CONTINUE
      DO 25 I=1,F
      LI=Q*(I-1)
      DO 25 J=1,Q
      X(LB+Q*N+LI+J)=ZDUM(J,D+I-1)
25    CONTINUE
C
C     CALCULATE NEW PARAMETER COVARIANCE MATRIX
C
      DO 30 I=1,LF
      ALPHA(I)=0.0
      DO 30 J=1,LF
      ALPHA(I)=ALPHA(I)+R(I,J)*X(J)
30    CONTINUE
      XO=0.0
      DO 40 I=1,LF
```

```
      XO=XO+X(I)*ALPHA(I)
      DO 40 J=1,LF
      BETA(I,J)=ALPHA(I)*ALPHA(J)
40    CONTINUE
      XO=XO+W*W
      DO 50 I=1,LF
      DO 50 J=1,LF
      R(I,J)=(R(I,J)-BETA(I,J)/XO)/(W*W)
50    CONTINUE
C
C     CALCULATE NEW PARAMETERS
C
      DO 60 I=1,Q
      GAMMA(I)=0.0
      ZZ(I)=0.0
      DO 70 J=1,LF
      GAMMA(I)=GAMMA(I)+P(I,J)*X(J)
70    CONTINUE
      DO 75 IL=1,Q
      ZZ(I)=ZZ(I)+PO(I,IL)*UDUM(IL,D)
75    CONTINUE
      GAMMA(I)=Y(I)-GAMMA(I)-ZZ(I)
      DO 60 J=1,LF
      DELTA(I,J)=GAMMA(I)*X(J)
      EPS(I,J)=0.0
60    CONTINUE
      DO 80 I=1,Q
      DO 80 J=1,LF
      DO 90 JJ=1,LF
      EPS(I,J)=EPS(I,J)+DELTA(I,JJ)*R(JJ,J)
90    CONTINUE
      P(I,J)=P(I,J)+EPS(I,J)
80    CONTINUE
C
C     CALCULATE INPUT SIGNALS
C
      DO 100 I=1,LS
      LI=Q*(I-1)
      DO 100 J=1,Q
      X(LI+J)=UDUM(J,I)
      X(LS*Q+J)=Y(J)
100   CONTINUE
      NN=N-1
```

```
      DO 110 I=1,NN
      LI=Q*I
      DO 110 J=1,Q
      X(LS*Q+LI+J)=YDUM(J,I)
110   CONTINUE
      DO 115 J=1,Q
      X((LS+N)*Q+J)=Z(J)
115   CONTINUE
      FF=F-1
      DO 116 I=1,FF
      LI=Q*I
      DO 116 J=1,Q
      X((LS+N)*Q+LI+J)=ZDUM(J,I)
116   CONTINUE
      DO 120 I=1,Q
      PSI(I)=0.0
      DO 130 J=1,LF
      PSI(I)=PSI(I)+P(I,J)*X(J)
130   CONTINUE
      PSI(I)=YRV(I,NFC+D)-PSI(I)
      DO 120 J=1,Q
      PHI(I,J)=PO(I,J)
120   CONTINUE
C
C     CALL SYSTEM SUBROUTINE TO CALCULATE
C     INVERSE MATRIX BETA(0)
C
      CALL MINV(PHI,2,DET,HELP1,HELP2)
      DO 150 I=1,Q
      U(I)=0.0
      DO 140 J=1,Q
      U(I)=(U(I)+PHI(I,J)*PSI(J))
140   CONTINUE
C
C     PUT CONSTRAINTS ON INPUT SIGNALS
C
159   IF(U(I).GE.20.0) U(I)=20.0
      IF(U(I).LE.-20.0) U(I)=-20.0
150   CONTINUE
      RETURN
      END
```

```
C     ****************************************************
C
C          SUBROUTINE CONTRL(REI,STI)
C
C        THIS SUBROUTINE IS A LINK BETWEEN THE SIMULATION
C     PROGRAM OF THE BINARY DISTILLATION COLUMN AND THE
C     MULTIVARIABLE SELF-TUNING CONTROLLER.
C     AT THE FIRST CALL, THE FOLLOWING PARAMETERS MUST
C     BE PROVIDED:
C        Q = NUMBER OF LOOPS TO BE CONTROLLED
C        D = ESTIMATED TIME DELAY + 1
C        N = NUMBER OF A MATRICES IN THE MODEL
C        M = NUMBER OF B MATRICES IN THE MODEL - 1
C        F = NUMBER OF MATRICES IN DISTURBANCE MODEL
C        R(I,I) = DIAGONAL ELEMENTS OF INITIAL PARAMETER
C                 COVARIANCE MATRIX
C        P(I,J) = INITIAL PARAMETER ESTIMATES (FIRST MATRIX
C                 MUST BE NONSINGULAR)
C        LAMDA(I,I) = NOISE MULTIPLICATOR
C        NF = COUNTER USED IF AVERAGE OUTPUT IS CONTROLLED
C        IRAFT = 0 OUTPUT CONTROLLED, SET POINT CONSTANT
C                1 AVERAGE OUTPUT CONTROLLED, SET POINT
C                IS VARIABLE
C        FORG = FORGETTING FACTOR
C
C     THE OTHER PARAMETERS ARE INITIALIZED IN A BLOCK
C     DATA ROUTINE.
C
C     ****************************************************
C
      SUBROUTINE CONTRL(REI,STI)
C
C     DIMENSIONALIZATION
C
      COMMON/AREA1/T,DT,NI,KK,JM,JN,MTT,LF
      COMMON/AREA2/XT(10),YT(10),LT(10),VT(10),WT(10),
     #HLT(10),HVT(10),TL(10),DWT(10),DHL(10),HLTO(10),
     #RX1,RX2
      COMMON/AREA3/ICHNG(10),DIST(10),FE,ST,RE,DF,IDT,ITYPE
      COMMON/AREA4/IEQ,E(10),EE(10),EF(10),QLP(10),QRP(10),
C     #QSP(10)
      COMMON/AREA5/HF,HR,QR,HSI,TR,TS,UA,XIN,XIF,XF
      COMMON/AREA6/ICON,CHAN(2),XDSP,XBSP
      COMMON/AREA7/XS(26),YS(26),TLS(26),XST(19),YST(19),
     #HXS(19),HYS(19)
      COMMON/AREA8/B1(26),C1(26),D1(26),B2(19),C2(19),
     #D2(19),B3(19),C3(19),D3(19)
      COMMON/AREA9/ Y(5),Z(5),U(5),EN(5),YR(5),AIE(5),
     #RAIE(5),DU(5)
      COMMON/AREA10/ YDUM(5,5),UDUM(5,10),DUDUM(5,10),
     #LAMDA(5,5),SA(5),ZDUM(5,5),IX(5)
      COMMON/AREA11/ X(60),P(5,60),R(60,60),PO(5,5),
```

```
     #YRV(5,200),YA(5)
      REAL LAMDA,SA,FORG
      INTEGER Q,F,D,IX
C
C     INITIALIZATION; ONLY DONE AT THE FIRST CALL TO THIS
C     PROGRAM
C
      IF(KK.GT.1) GOTO 10
      U(1)=REI
      U(2)=STI
      READ(5,1) Q,D,N,M,F
      LLEF=Q*(N+F+M+D+1)
      DO 5 I=1,Q
      READ(5,2) (P(I,J),J=1,LLEF)
5     CONTINUE
      READ(5,2) (R(I,I),I=1,LLEF)
      READ(5,2) (LAMDA(I,I),I=1,Q)
      READ(5,1) NF,IRAFT
      READ(5,2) FORG
      NFC=0
1     FORMAT(5I3)
2     FORMAT(10F16.3)
3     FORMAT(16F8.3)
      DO 20 I=1,Q
      AIE(I)=0.0
      RAIE(I)=0.0
      IX(I)=333+17*I
      SA(I)=0.0005
20    CONTINUE
      AM=0.0
10    CONTINUE
C
C     VARIABLES FROM COLUMN SIMULATION ARE TRANSFERRED TO
C     APPROPRIATE VECTORS
C
      Y(1)=XT(MTT)
      Y(2)=XT(1)
      Z(1)=FE
      Z(2)=FE
      YR(1)=XDSP
      YR(2)=XBSP
      K=D-1
C
C     NOISE IS ADDED TO THE OUTPUT SIGNALS
C
      IF(NFC.EQ.NF) NFC=0
      NFC=NFC+1
      DO 35 I=1,Q
      CALL GAUSS(IX(I),SA(I),AM,EN(I))
35    CONTINUE
      DO 30 I=1,Q
      DO 40 J=1,Q
```

```
      Y(I)=Y(I)+LAMDA(I,J)*EN(J)
40    CONTINUE
C
C     OUTPUT AND FEED SIGNALS ARE NORMALIZED
C
      Y(I)=(Y(I)-YR(I))/YR(I)
      Z(I)=(Z(I)-18.055)/18.055
C
C     CALCULATION OF OUTPUT AVERAGES AND VARIANCES
C
      AIE(I)=AIE(I)+Y(I)*Y(I)
      YA(I)=YA(I)+Y(I)
      YRV(I,NFC+D)=0.0
C
C     IF OUTPUT AVERAGE IS CONTROLLED, CALCULATE THE NEW
C     SET POINT
C
      IF(IRAFT.EQ.0.OR.NFC.GE.NF+1-D) GOTO 12
      LM1=NFC+1
      LM2=NFC+D-1
      DO 50 J=LM1,LM2
      YRV(I,NFC+D)=YRV(I,NFC+D)+YRV(I,J)
50    CONTINUE
      YRV(I,NFC+D)=(-YRV(I,NFC+D)+YR(I)*NF-YA(I))/
     #(NF+1-NFC-D)
      GOTO 30
12    YRV(I,NFC+D)=YR(I)
30    CONTINUE
C
C     CALL MULTIVARIABLE SELF-TUNING REGULATOR PROGRAM
C
      CALL MSTR(N,M,F,D,Q,FORG,DUDUM,YDUM,ZDUM,X,R,Y,Z,YRV
     #,DU,P,NFC,PO,KK)
C
C     SHIFT VARIABLES IN VECTORS USED IN STR-ALGORITHM,
C     TEST INPUT FOR VALUES LOWER THAN ZERO AND
C     CALCULATE CHANGE IN INPUT SIGNAL FOR THE COLUMN
C
      DO 70 J=1,9
      JJ=10-J
      DO 70 I=1,Q
      UDUM(I,JJ+1)=UDUM(I,JJ)
      DUDUM(I,JJ+1)=DUDUM(I,JJ)
70    CONTINUE
      CHAN(1)=REI*DU(1)
      CHAN(2)=STI*DU(2)
      U(1)=U(1)+CHAN(1)
      U(2)=U(2)+CHAN(2)
      GOTO 77
78    IF(U(1).GT.0.0) GOTO 76
      U(1)=0.0
      DU(1)=0.0
```

```
76     IF(U(2).GT.0.0) GOTO 77
       U(2)=0.0
       DU(2)=0.0
77     CONTINUE
       DO 80 J=1,4
       JJ=5-J
       DO 80 I=1,Q
       YDUM(I,JJ+1)=YDUM(I,JJ)
       ZDUM(I,JJ+1)=ZDUM(I,JJ)
80     CONTINUE
       DO 90 I=1,Q
       YDUM(I,1)=-Y(I)
       UDUM(I,1)=U(I)
       ZDUM(I,1)=Z(I)
       DUDUM(I,1)=DU(I)
90     CONTINUE
C
C      PRINT OUTPUTS, INPUTS, PARAMETERS AND CALCULATED
C      VARIABLES
C
24     FORMAT(1H ,20(E12.5,1X))
       WRITE(9,24) XT(MTT),XT(1),AIE(1),AIE(2),R(1,1),R(2,2)
      #,FORG
25     FORMAT(40F12.8)
       DO 26 I=1,Q
       WRITE(8,25) (P(I,J),J=1,LLEF)
26     CONTINUE
       IF(NFC.LT.NF) GOTO 500
C
C      RESET AVERAGE AFTER SPECIFIED TIME PERIOD NF
C
       DO 1100 I=1,Q
       RAIE(I)=(YA(I)-YR(I)*NF)*(YA(I)-YR(I)*NF)/(NF*NF)
       YA(I)=0.0
       JJ=NF+D
       DO 1100 J=1,JJ
       YRV(I,J)=0.0
1100   CONTINUE
500    CONTINUE
800    CONTINUE
600    RETURN
       END
```

```
C       ******************************************************
C
C       BLOCK DATA FOR INITIALIZATION OF PARAMETERS
C
C       ******************************************************
 C
        BLOCK DATA
        COMMON/AREA9/Y(5),Z(5),U(5),EN(5),YR(5),AIE(5),RAIE(5)
       #,DU(5)
        COMMON/AREA10/YDUM(5,5),UDUM(5,10),DUDUM(5,10),
       #LAMDA(5,5),SA(5),ZDUM(5,5),IX(5)
        COMMON/AREA11/X(60),P(5,60),R(60,60),PO(5,5),
       #YRV(5,200),YA(5)
        REAL LAMDA
        DATA Y,Z,U,EN,YR,AIE,RAIE,DU,SA,IX/45*0.0,5*0/
        DATA YDUM,UDUM,DUDUM,LAMDA,ZDUM/175*0.0/
        DATA X,P,R,YRV,YA/4965*0.0/
        END
```

```
C     *********************************************************
C
C          SUBROUTINE MSTR
C
C        THIS SUBROUTINE CALCULATES A MULTIVARIABLE
C     SELF-TUNING CONTROLLER WITH INTEGRAL CONTROL AND
C     FEEDFORWARD COMPENSATION. THE ROUTINE IS INDEPENDENT
C     OF ITS VARIABLES AND CAN BE CALLED FROM DIFFERENT
C     PROGRAMS. ALL PARAMETERS ARE PASSED TO THIS ROUTINE
C     FROM THE LINKING PROGRAM.
C     ALL PARAMETERS ARE ESTIMATED AND IF THE NEW
C     ESTIMATE OF B0 IS SINGULAR, THE PREVIOUS VALUE
C     IS RETAINED.
C
C     N = NUMBER OF A MATRICES IN THE MODEL
C     M = NUMBER OF B MATRICES IN MODEL - 1
C     F = NUMBER OF MATRICES IN DISTURBANCE MODEL
C     D = TIME DELAY + 1
C     Q = NUMBER OF LOOPS
C     W = FORGETTING FACTOR
C     UDUM = MATRIX TO STORE PAST INPUT CHANGES
C     YDUM = MATRIX TO STORE PAST OUTPUT SIGNALS
C     ZDUM = MATRIX TO STORE PAST DISTURBANCE SIGNALS
C     X = MATRIX PSI IN STR ALGORITHM
C     R = PARAMETER COVARIANCE MATRIX
C     Y = OUTPUT SIGNALS
C     Z = DISTURBANCE SIGNALS
C     YRV = REFERENCE OUTPUT
C     U = INPUT CHANGES
C     P = PARAMETER MATRIX
C     NFC = COUNTER
C     PO = FIRST B PARAMETER MATRIX
C     KK = COUNTER
C
C     *********************************************************
C
      SUBROUTINE MSTR(N,M,F,D,Q,W,UDUM,YDUM,ZDUM,X,R,Y,Z
     #,YRV,U,P,NFC,PO,KK)
      DIMENSION PO(5,5),ZZ(5),X(60),UDUM(5,10),YDUM(5,5),
     #ALPHA(60),R(60,60),ZDUM(5,5)
      DIMENSION BETA(60,60),Z(5),GAMMA(5),Y(5),U(5),
     #DELTA(5,60),PSI(5)
      DIMENSION EPS(5,60),P(5,60),YRV(5,200),PHI(2,2)
      INTEGER Q,F,FF,D
C
C     CALCULATE INTERNAL COUNTERS
C
      LLEF=Q*(N+F+M+D+1)
      LS=M+D
C
```

```
C     FORM MATRIX WITH PAST Y, U, Z VALUES
C
      DO 10 I=1,LS
      LI=Q*(I-1)
      DO 10 J=1,Q
      X(LI+J)=UDUM(J,D+I-1)
10    CONTINUE
      LB=LS*Q
      NN=N+1
      DO 20 I=1,NN
      LI=Q*(I-1)
      DO 20 J=1,Q
      X(LB+LI+J)=YDUM(J,D+I-1)
20    CONTINUE
      DO 25 I=1,F
      LI=Q*(I-1)
      DO 25 J=1,Q
      X(LB+Q*NN+LI+J)=ZDUM(J,D+I-1)
25    CONTINUE
C
C     CALCULATE NEW COVARIANCE MATRIX
C
      DO 30 I=1,LLEF
      ALPHA(I)=0.0
      DO 30 J=1,LLEF
      ALPHA(I)=ALPHA(I)+R(I,J)*X(J)
30    CONTINUE
      XO=0.0
      DO 40 I=1,LLEF
      XO=XO+X(I)*ALPHA(I)
      DO 40 J=1,LLEF
      BETA(I,J)=ALPHA(I)*ALPHA(J)
40    CONTINUE
      XO=XO+W*W
      DO 50 I=1,LLEF
      DO 50 J=1,LLEF
      R(I,J)=(R(I,J)-BETA(I,J)/XO)/(W*W)
50    CONTINUE
C
C     CALCULATE PREDICTION ERROR
C
      DO 60 I=1,Q
      GAMMA(I)=0.0
      ZZ(I)=0.0
      DO 70 J=1,LLEF
      GAMMA(I)=GAMMA(I)+P(I,J)*X(J)
70    CONTINUE
      GAMMA(I)=Y(I)-GAMMA(I)
      DO 60 J=1,LLEF
      DELTA(I,J)=GAMMA(I)*X(J)
      EPS(I,J)=0.0
60    CONTINUE
      DO 80 I=1,Q
```

```
      DO 80 J=1,LLEF
      DO 90 JJ=1,LLEF
      EPS(I,J)=EPS(I,J)+DELTA(I,JJ)*R(JJ,J)
90    CONTINUE
C
C     CALCULATE PARAMETER UPDATES
C
      P(I,J)=P(I,J)+EPS(I,J)
80    CONTINUE
      LSS=M+D-1
C
C     FORM NEW MATRIX WITH OLD Y, U, Z VALUES
C
      DO 100 I=1,LSS
      LI=Q*(I-1)
      DO 100 J=1,Q
      X(LI+J)=UDUM(J,I)
      X(LSS*Q+J)=-Y(J)
100   CONTINUE
      DO 110 I=1,N
      LI=Q*I
      DO 110 J=1,Q
      X(LSS*Q+LI+J)=YDUM(J,I)
110   CONTINUE
      DO 115 J=1,Q
      X((LSS+1+N)*Q+J)=Z(J)
115   CONTINUE
      FF=F-1
      DO 116 I=1,FF
      LI=Q*I
      DO 116 J=1,Q
      X((LSS+1+N)*Q+LI+J)=ZDUM(J,I)
116   CONTINUE
C
C     CALCULATE PREDICTED OUTPUT
C
      LEFF=Q*(N+F+M+D)
      DO 120 I=1,Q
      PSI(I)=0.0
      DO 130 J=1,LEFF
      JJ=J+Q
      PSI(I)=PSI(I)+P(I,JJ)*X(J)
130   CONTINUE
      PSI(I)=-PSI(I)
120   CONTINUE
C
C     CHECK IF B0 NONSINGULAR
C
      PDET=P(1,1)*P(2,2)-P(1,2)*P(2,1)
```

```
      IF(PDET.EQ.0.0) GOTO 122
      PHI(1,1)=P(2,2)/PDET
      PHI(2,2)=P(1,1)/PDET
      PHI(1,2)=-P(1,2)/PDET
      PHI(2,1)=-P(2,1)/PDET
C
C     CALCULATE CHANGE IN INPUT SIGNALS
C
122   DO 150 I=1,Q
      U(I)=0.0
      DO 140 J=1,Q
      U(I)=U(I)+PHI(I,J)*PSI(J)
140   CONTINUE
C
C     CONSTRAIN INPUT CHANGES
C
      IF(U(1).GE.0.2) U(1)=0.2
      IF(U(1).LE.-0.2) U(1)=-0.2
      IF(U(2).GE.0.2) U(2)=0.2
      IF(U(2).LE.-0.2) U(2)=-0.2
150   CONTINUE
      RETURN
      END
```

```
C       ******************************************************
C
C          SUBROUTINE GAUSS GENERATES A RANDOM NOISE SIGNAL
C       WITH MEAN AM AND VARIANCE SAA.
C
C       ******************************************************
C
        SUBROUTINE GAUSS(IXI,SAA,AM,V)
        AA=0.0
        DO 7 I=1,12
        IY=IXI*65539
        IF(IY) 5,6,6
5       IY=IY+2147483647+1
6       YFL=IY
        YFL=YFL*0.46566133E-9
        IXI=IY
7       AA=AA+YFL
        V=(AA-6.0)*SAA+AM
        RETURN
        END
```